CHAPTER 1

HOW COMPUTERS WORK

In this chapter, you will learn:

- About the functions performed by different hardware components of a microcomputer
- About the three kinds of software and how they relate to one another and to hardware
- How the CPU uses primary and secondary storage to manage software

Like millions of other computer users, you have probably used your microcomputer to play games, explore the Internet, write papers, build spreadsheets, or create a professional-looking proposal or flyer. You can perform all these applications without understanding exactly what goes on behind your computer case or monitor screen. But if you are curious to learn more about microcomputers, and if you want to graduate from simply being the end user of your computer to becoming the master of your machine, then this book is for you. This book is written for anyone who wants to understand what is happening "behind the scenes" in order to install and set up new software, install new hardware, diagnose both hardware and software problems, and make decisions about purchasing new hardware.

This chapter introduces you to the inside of your computer, a world of electronic and mechanical devices that has evolved over just a few years to become one of the most powerful technical tools of our society. The only assumption made here is that you are a computer user—that is, you can turn on your machine, load a software package, and use that software to accomplish a task. This book will help you see what goes on behind the scenes when you do those things.

In the world of computers, the term **hardware** refers to the physical components of the computer, such as the monitor, keyboard, memory chips, and hard drive. The term **software** refers to the set of instructions that directs the hardware to accomplish a task. In Figure 1-1, a passenger directs a chauffeur who directs a car, just as a computer user directs software which, in turn, controls hardware to perform a given task.

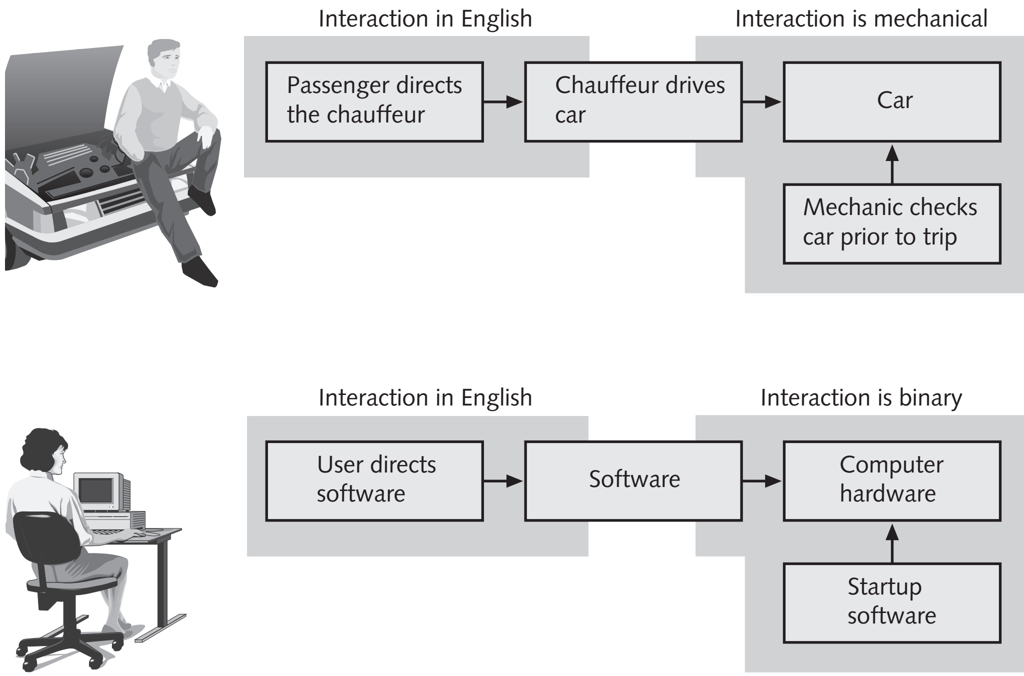

Figure 1-1 A user interacts with a computer much as a passenger interacts with a chauffeured car

In order to perform a computing task, hardware is used by software for four basic functions: input, processing, output, and storage (see Figure 1-2). Also, hardware components must communicate both data and instructions among themselves, and, since these components are electrical, an electrical system is required. In this chapter, we introduce the hardware and software components of a computer system. In Chapter 2 we address how they work together, with the primary focus on the sophisticated system of communication of data and instructions that includes both hardware and software.

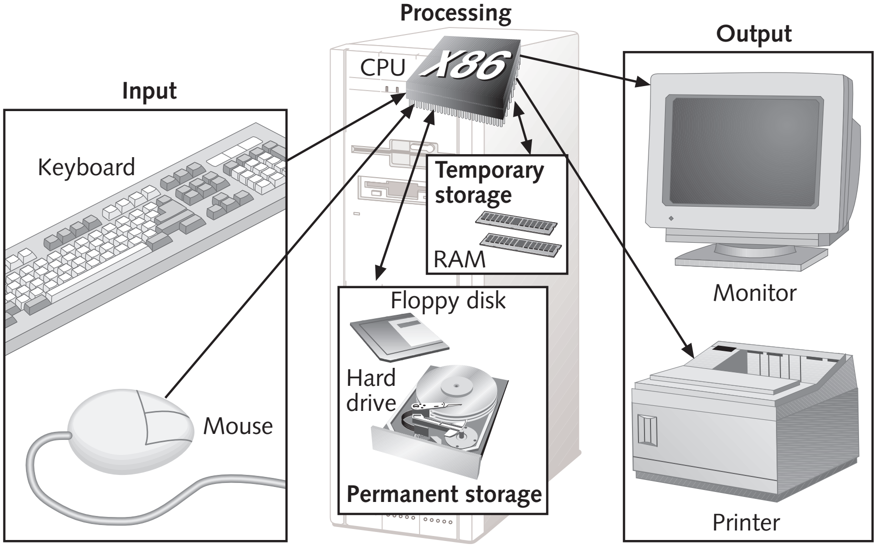

Figure 1-2 The CPU is central to all computer activity, which consists of input, processing, storage, and output

Looking again at Figure 1-1, the interaction between the passenger and the chauffeur is in English, but the chauffeur must translate these directions such as, "Go to the airport," into mechanical directions that the car can "understand" such as pressing the gas pedal, braking when necessary, and turning the wheel to control the car until it reaches the airport. In the same fashion, a computer user interacts with a computer in a language that the user understands, but software must convert that instruction into a form that hardware can "understand" (see Figure 1-3). Hardware stores data and communicates with software by only one fundamental method—binary—and, in effect, speaks a language that only has two words, "on" and "off." Every communication that software has with hardware is reduced to a series of these two words.

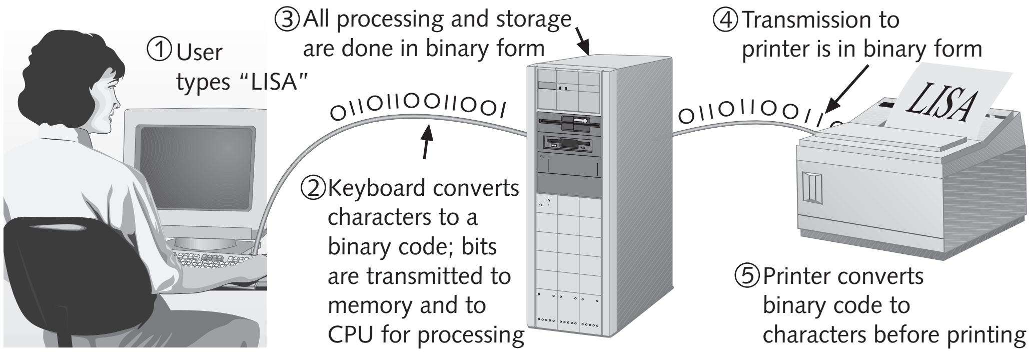

Figure 1-3 All communication, storage, and processing of data inside a computer are in binary until presented as output to the user

It was not always so. For almost half a century, people attempted to invent an electronic computational device that could store all 10 digits in our decimal number system and even some of our alphabet. No one could invent a device that could effectively store all 10 digits. It was not until the 1940s that a breakthrough occurred when it was suggested that only two numbers be stored, reducing all counting and calculations to a number system that only needed two digits, 0 and 1. The technology to hold only two states, on and off, existed. Only two voltage levels were required for a functional switch, zero volts or some volts. Appendix D discusses this **binary number system**, which is the language of a computer. When you study how computers work, this one fact permeates everything, and keeping this binary concept in mind makes everything much clearer. A computer's world is a binary world, and communication of instructions and data by the devices that process them is always in binary.

TIP

Every communication, every process (including the storage of data and instructions) in a computer is a series of zeros and ones.

HARDWARE

Hardware without software is like a car with no driver; it's useless. So when we examine different hardware devices, we really don't see how to use them until we introduce software. In this section we cover the major hardware components of a microcomputer system used for input, output, processing, storage, electrical supply, and communication. Later in this chapter, we'll discuss how software is used to make the hardware work for us.

A+CORE 1.1

Most input and output devices are outside the computer case, and most processing and storage are done inside the case. The central, most important hardware device in a computer is the **central processing unit** (**CPU**), the **microprocessor** or **processor**. As its name implies, this device is central to all processing done by the computer. Data received by input devices goes to the CPU, and output travels from the CPU to output devices. The CPU stores data and instructions in storage devices and performs calculations and other processing of data as well. Whether inside or outside the case, and regardless of the function the device performs, each device requires these things to operate:

- **A method for the CPU to communicate with the device.** The device must send data to and/or receive data from the CPU. The CPU might need to control the device by passing instructions to it, and/or the device might need to request service from the CPU.
- **Software to instruct and control the device.** A device is useless without software to control it. The software must know how to communicate with the device at the detailed level of that specific device, and the CPU must have access to this software in order to interact with the device.

- **Electricity to power the device.** Electronic devices require electricity to operate. Devices can either receive power from the power supply inside the computer case, or they can have their own power supply by way of a power cable to an electrical outlet.

In the next few pages, we take a sightseeing tour of computer hardware, first looking outside and then inside the case.

Hardware Used for Input and Output

A+CORE 1.2

Most input/output devices reside outside the computer case. These devices communicate with what is inside the computer case through cables attached to the case at a connection called a **port**, sending data and/or instructions to the computer and receiving them from the computer. Most computers have their ports located on the back of the case (Figure 1-4), but some models put the ports on the front of the case for easy access. The most popular input devices are a keyboard and a mouse, and the most popular output devices are a monitor and a printer.

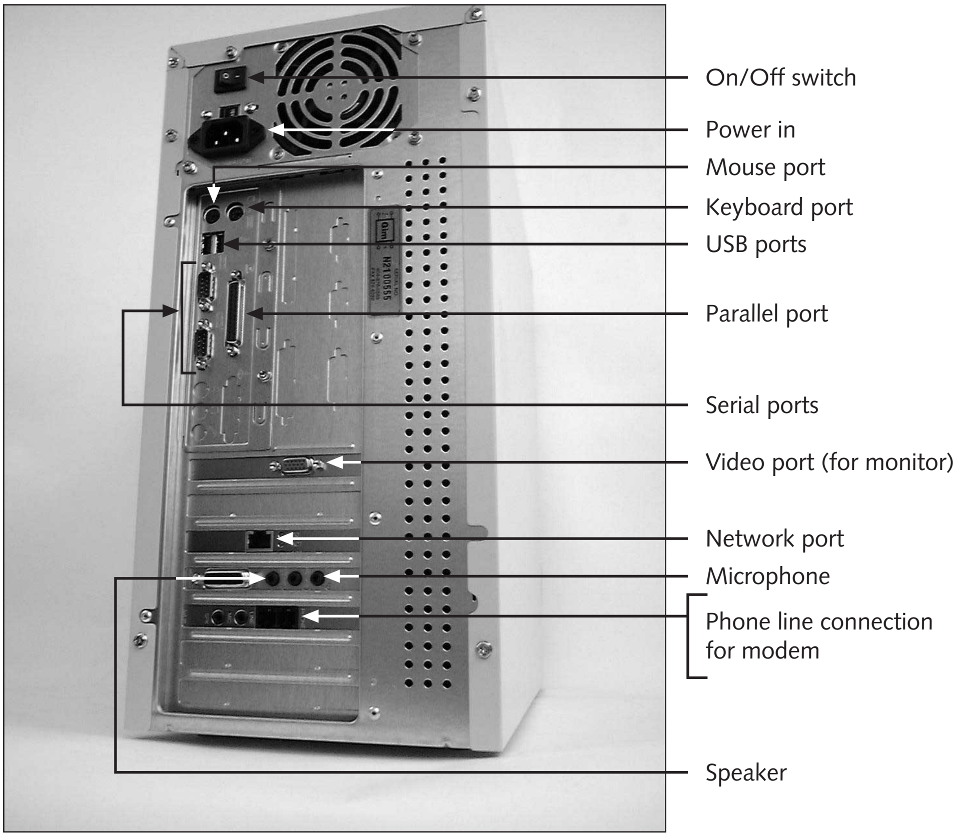

Figure 1-4 Input/output devices connect to the computer case by ports usually found on the back of the case

The **keyboard** is the primary input device of a computer (see Figure 1-5). The keyboards that are standard today are called enhanced keyboards and hold 102 keys. Some keyboards are curved to be more comfortable for the hands and wrists, and are called ergonomic keyboards. In addition, some keyboards come equipped with a mouse port—a plug into which a mouse (another input device) can be attached to the keyboard—although it is more common for the mouse port to be located directly on the computer case. Electricity to run the keyboard comes from inside the computer case and is provided by wires in the keyboard cable.

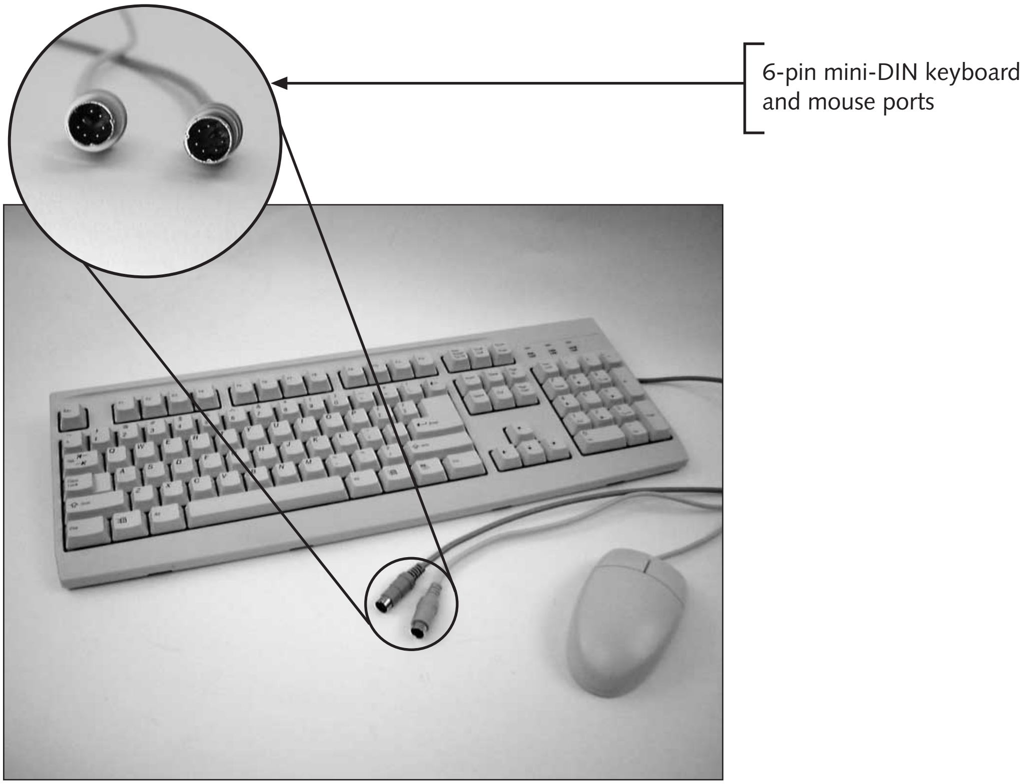

Figure 1-5 The keyboard and the mouse are the two most popular input devices

A **mouse** is a pointing device used to move a pointer on the screen and to make selections. The bottom of a mouse houses a rotating ball that tracks movement and controls the location of the pointer. The one, two, or three buttons on the top of the mouse serve different purposes for different software. For example, Windows 98 uses the left mouse button to execute a command and the right mouse button to display information about the command.

Both the keyboard and the mouse receive input mechanically (you press a key or move the mouse), and this movement is converted into binary data that is input into the computer.

The monitor and the printer are the two most popular output devices (see Figure 1-6). The **monitor** is the visual device that displays the primary output of the computer. Once, all monitors were monochrome (one color), but today they display text and graphics in color.

Hardware manufacturers typically rate a monitor according to the size of its screen (in inches) and by the number of dots on the screen used for display. A **pixel** is a dot or unit of color that is the smallest unit of display on a monitor.

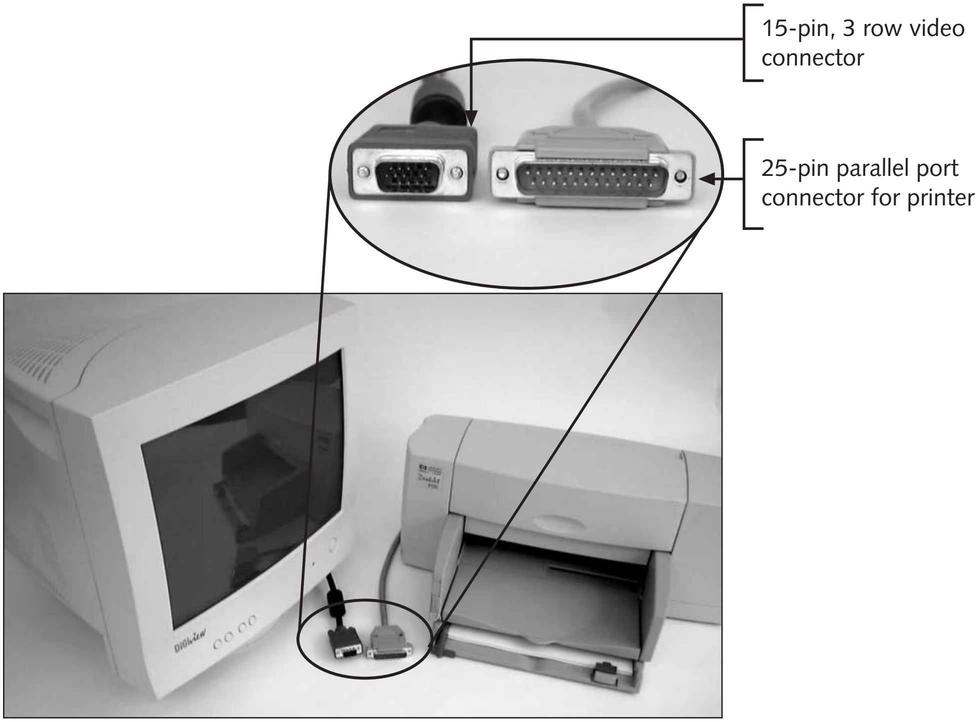

Figure 1-6 The two most popular output devices are the monitor and the printer

A very important output device is the **printer**, which produces output on paper, often called **hard copy**. The most popular printers available today are ink-jet, laser, and dot matrix printers. The monitor and the printer each needs its own power supply. Their electrical power cords connect to electrical outlets. Sometimes the computer case provides an electrical outlet for the monitor's power cord to eliminate the need for one more power outlet.

Hardware Inside the Computer Case

Most storage and all processing of data and instructions are done inside the computer case, so before we look at components used for storage and processing, let's look at what you see when you first open the computer case. Most computers contain these devices inside the case (see Figure 1-7):

- A system board containing the CPU, memory, and other components
- A floppy drive, hard drive, and CD-ROM drive used for permanent storage

- A power supply with power cables supplying electricity to all devices inside the case
- Circuit boards used by the CPU to communicate with devices inside and outside the case
- Cables connecting devices to circuit boards and the system board

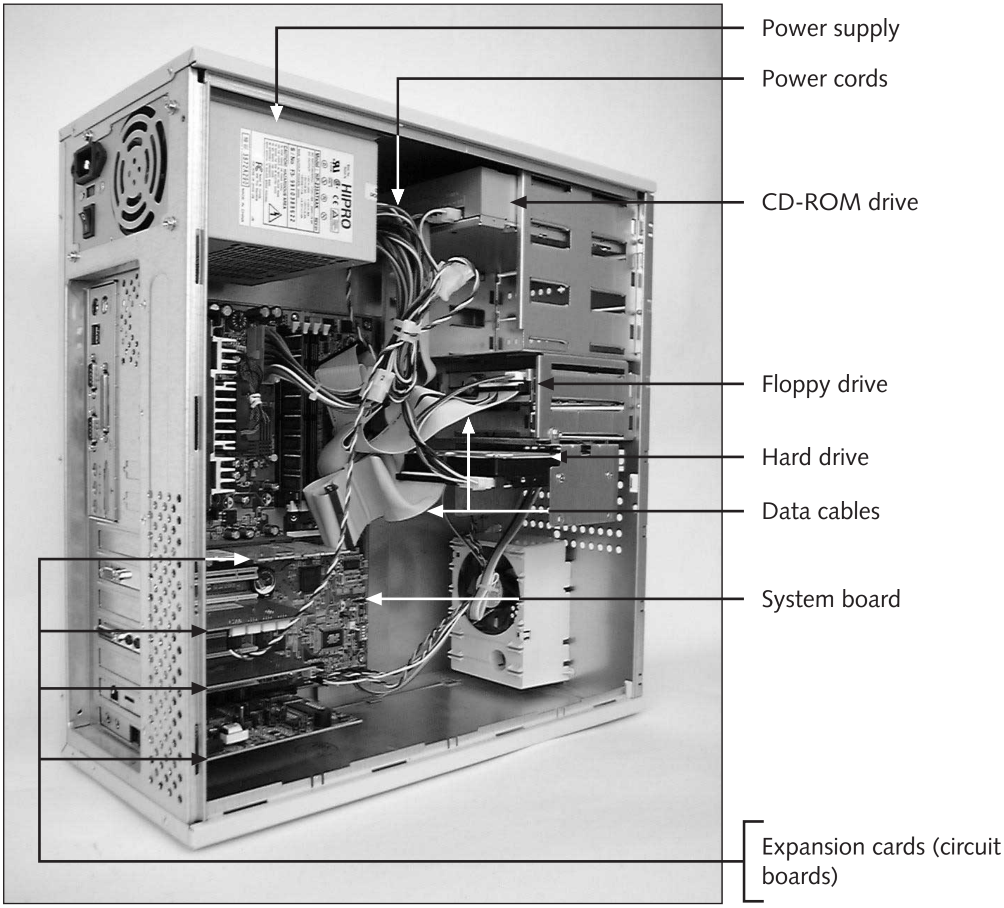

Figure 1-7 Inside the computer case

A+ CORE 1.1

Among the things you first notice when you look inside a computer case are boards that contain electronic components. A **circuit board** is a board that holds microchips or integrated circuits (ICs) and the circuitry that connects these chips. All circuit boards contain microchips, which are manufactured in one of two ways: CMOS (complementary metal-oxide semiconductor) chips or TTL (transistor-transistor logic) chips. CMOS chips require less electricity, hold data longer after the electricity is turned off, are slower, and produce less heat than TTL chips do. Most CPUs are CMOS chips.

The other major components look like small boxes, including the power supply, hard drive, CD-ROM drive, and floppy drive. Devices that the CPU communicates with that are not located directly on the system board are called **peripheral devices** and are linked to the CPU through a connection to the system board. Some peripheral devices are linked to the system board by a circuit board designed for that purpose. These circuit boards, called **expansion cards**, are installed in long narrow slots on the system board called **expansion slots**.

There are two types of cables inside the case: data cables, which connect devices to one another, and power cables, which supply power from the power supply. Most often, you can distinguish between the two by the shape of the cable. Data cables are flat and wide, and power cables are round and small. There are some exceptions to this rule, so the best way to identify a cable is to trace its source and destination. All power cables originate from the power supply.

A+CORE 1.1

The System Board

The largest and most important circuit board in the computer is the **system board**, **mainboard**, or **motherboard** (see Figure 1-8), which contains the CPU, the component where most processing takes place. The system board is the most complicated piece of equipment inside the case and is covered in detail in Chapter 3. Because all devices must communicate with the CPU on the system board, all devices in a computer are either installed directly on the system board, linked to it by a cable connected to a port on the system board, or indirectly linked to it by expansion cards. Some ports on the system board stick outside the case to accommodate external devices such as a keyboard, and some ports provide a connection for a device inside the case, such as a floppy disk drive.

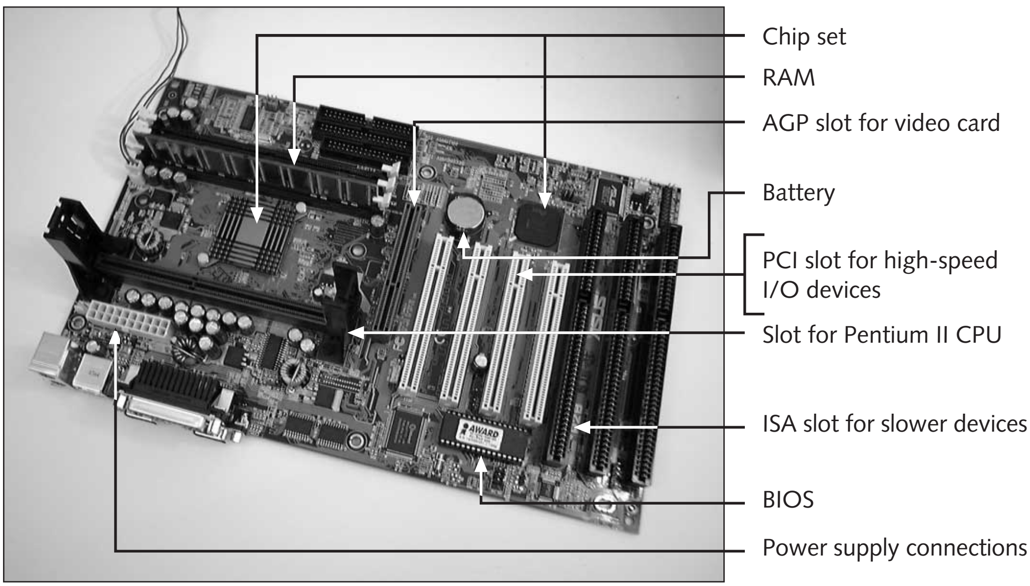

Figure 1-8 A Pentium system board. All hardware components are either located on the system board or directly or indirectly connected to it, because they must all communicate with the CPU.

Listed below are the major components found on all system boards, some of which are labeled in Figure 1-8 and are discussed in detail in the sections that follow.

Components used primarily for processing:

- Central processing unit (CPU), the computer's most important chip
- Chip set that supports the CPU by controlling many system board activities

Components used for temporary storage:

- Random access memory (RAM) used to hold data and instructions as they are processed
- Cache memory to speed up memory access (optional—depends on the type of CPU)

Components that allow the CPU to communicate with other devices:

- Bus used for communication on the system board
- Expansion slots to connect expansion cards to the system board

Firmware and setup information stored on the system board:

- Flash BIOS (basic input/output system) memory chip used to permanently store programs that control basic hardware functions (discussed in more detail later in the chapter)
- CMOS configuration chip

Electrical system:

- Power supply connections to provide electricity to the system board and expansion cards

Components Used Primarily for Processing

A+ CORE 1.1

The CPU or microprocessor is the chip inside the computer on which most computer processes must ultimately be executed (see Figure 1-9). Today most computers also contain microchips (called the **chip set**) that relieve the CPU of some low-level processing and provide careful timing of activities to increase the overall speed and performance of the computer. While this book will touch on different types of machines, it focuses on the most common personal computers (PCs), referred to as IBM-compatible, which are built around a family of microprocessors manufactured by Intel Corporation. The Macintosh family of computers, manufactured by Apple Computer, Inc., is built around a family of microprocessors manufactured by Motorola Corporation.

In addition to the CPU and the chip set, some older system boards also contain a chip that supports and enhances the function of some older CPUs. Many applications used this chip, called a **coprocessor**, to speed up the performance of certain CPU math functions. Most system boards manufactured before 1995 have a special socket for the coprocessor.

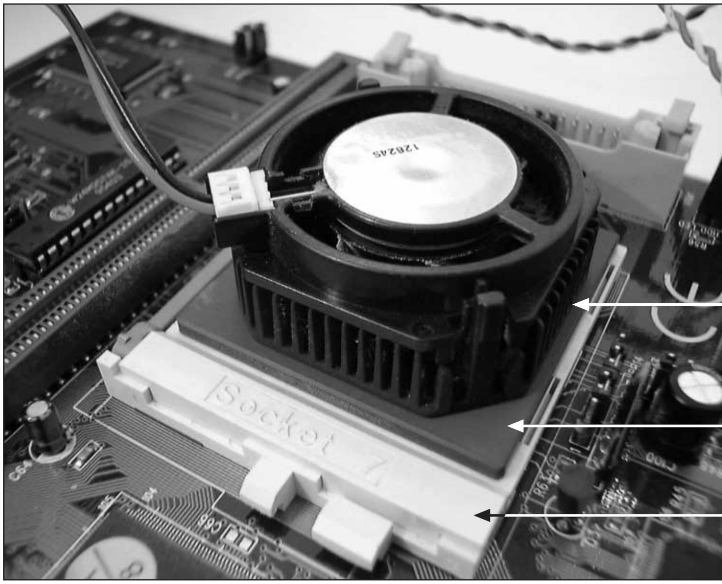

Figure 1-9 Processing of data and instructions is done by the CPU. This Pentium with fan on top is made by Intel.

Temporary (Primary) Storage Devices

A+ CORE 1.1

Figure 1-2 shows two kinds of storage: temporary and permanent. The CPU uses temporary storage, called **primary storage**, which is much faster to access than permanent storage, to temporarily hold both data and instructions while it is processing them. Primary storage is provided by devices on the system board and on other circuit boards, called memory or **random access memory** (**RAM**). RAM chips can be installed individually directly on the system board or in banks of several chips on a small board that plugs into the system board. The most common types of boards that hold memory chips are called **SIMMs** (**single inline memory modules**), **DIMMs** (**dual inline memory modules**), and **RIMMs** (see Figure 1-10). Whatever information is stored in primary storage is lost when the computer is turned off, because RAM chips need a continuous supply of electrical power to hold data or software stored in them. This kind of memory is called **volatile** because it is by nature temporary. By contrast, memory that holds its data permanently, such as that etched into ROM chips, is called **nonvolatile**.

Also part of primary storage is a special kind of very fast RAM called **cache memory**, which speeds up memory access by serving as a cache, or holding area, for data or instructions that are accessed frequently (see Figure 1-11). For newer CPUs, cache memory is stored inside the CPU housing on a memory chip that sits very close to the CPU microchip. For older CPUs, cache memory is stored on the system board either in individual chips or on memory modules called **COAST** (**cache on a stick**).

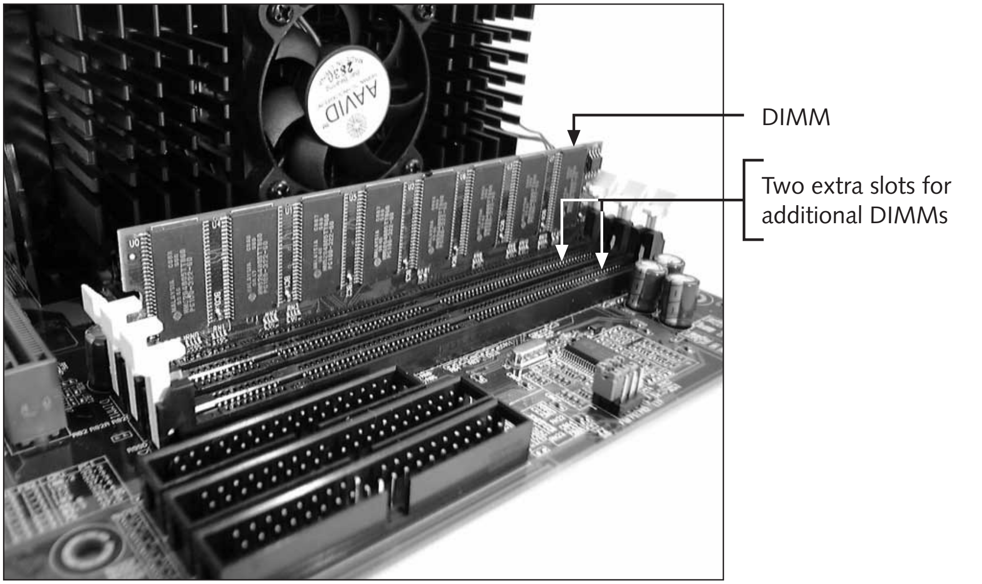

Figure 1-10 A SIMM or a DIMM holds RAM and is mounted directly on a system board

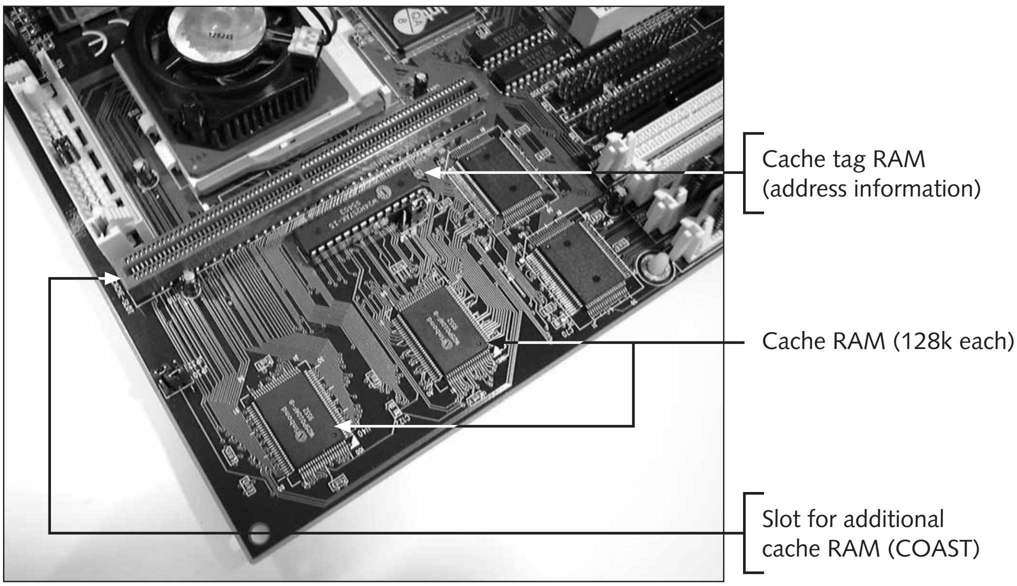

Figure 1-11 The speed of memory access is improved by using cache memory

Permanent (Secondary) Storage Devices

A+CORE 1.1

As you remember, the RAM on the system board is called primary storage. Primary storage temporarily holds both data and instructions as the CPU processes them. These data and instructions are also permanently stored on devices such as hard drives and floppy disks, in locations that are remote from the CPU. Data and instructions cannot be processed by the CPU from this remote storage (called **secondary storage**), but must first be copied into primary storage (RAM) for processing. The most important difference between these types of storage is that secondary storage is permanent. When you turn off your computer, the information in secondary storage remains intact. Conversely, the information in primary storage, or RAM, is lost when you turn off the machine. The four most popular secondary storage devices are hard disks, floppy disks, Zip drives, and CD-ROMs.

TIP

Don't forget that primary storage is temporary; as soon as you turn off the computer, any information there is lost. That's why you should always save your work frequently into secondary storage, which is permanent. There, your work remains safe even after the computer is turned off.

A **hard drive** is a sealed case containing platters or disks that rotate at a high speed (see Figure 1-12). As the platters rotate, an arm with a sensitive read/write head reaches across the platters, both writing new data to them and reading existing data from them. Managing a hard drive and understanding how one works are the subjects of Chapters 6 and 7. A hard drive requires a controller to manage it. The **hard drive controller** is a set of microchips that contain software to manage the hard drive, and temporarily hold data that is passed to and from the hard drive. In the past, hard drives using older technology required a separate controller expansion card. Today, a controller on a circuit board attaches directly to the hard drive. In this case, the data cable from the hard drive connects to the system board or to a small hard drive adapter card in an expansion slot. The adapter card is the interface between the hard drive and the CPU; data passes from the hard drive to the adapter card to the system board to the CPU. Most system boards today have a connection for the hard drive data cable on the system board (see Figure 1-13).

The older hard drives that had a controller on a large interface card in an expansion slot used two cables to connect the controller card to the hard drive. One cable was used to transmit instructions that controlled the hard drive from the controller card, and the other passed data back and forth. The controller and the hard drive had to use the same type of technology for communication to be possible.

Figure 1-12 Hard drive with sealed cover removed

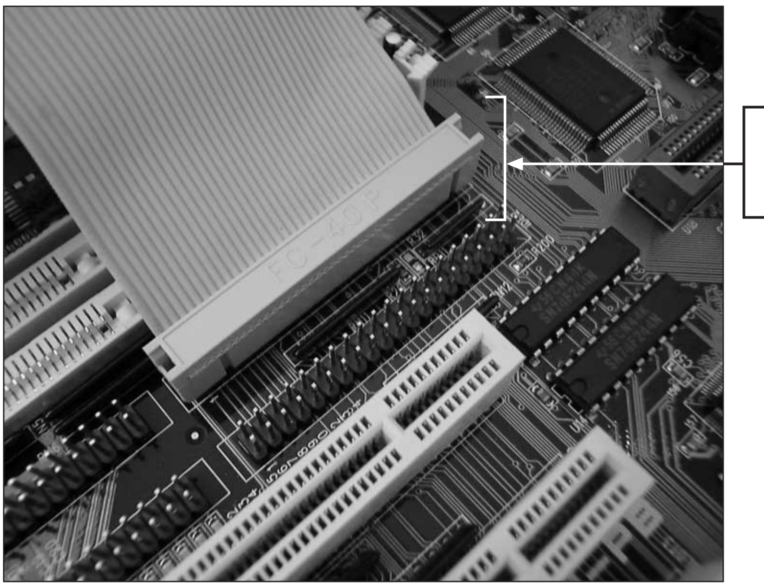

Figure 1-13 Most hard drives today connect to the system board by way of a data cable connected directly to the board

Another secondary storage device almost always found inside the case is a floppy drive. Floppy drives come in two common sizes: 3½ inches and 5¼ inches (referring to the size of the disks the drives can hold). The newer 3½-inch disks use more advanced technology and actually hold more data than the older 5¼-inch disks did. Like hard drives, floppy drives are managed by controllers. These controllers can be on interface cards, or can be chips on the system board. You can tell which your computer has by following the data cable from your floppy drive to the board.

A+CORE 1.1

Figure 1-14 shows a complete floppy drive subsystem in a microcomputer. A disk is inserted into the front of the floppy drive. Electricity to the drive is provided by a power cord from the power supply that connects to a power port at the back of the drive. A data cable also is connected to the back of the drive and runs either to a controller card inserted into an expansion slot on the system board or directly to the system board that contains controller chips, which completes the subsystem.

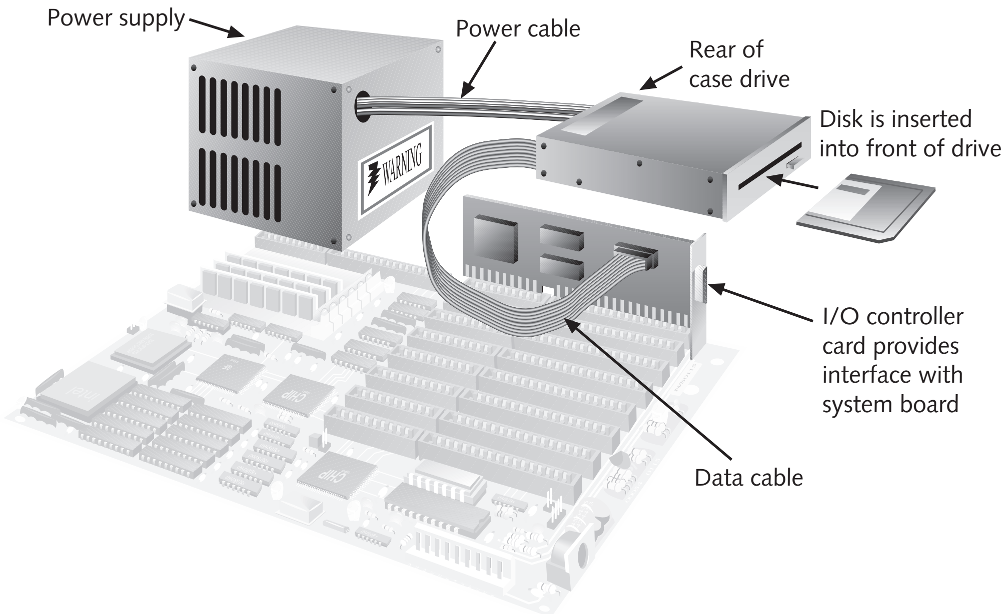

Figure 1-14 A floppy drive subsystem

Hard drives and floppy drives require power to operate. This power is received from the power supply through a power cord that connects at the rear of the drive.

System Board Components Used for Communication Among Devices

When you look carefully at the system board, you see many fine lines on both the top and the bottom of the board's surface (see Figure 1-15). These lines, sometimes called **traces**, are circuits, or paths, that enable data, instructions, and power to move from component to component on the board. This system of pathways used for communication and the protocol and

methods used for transmission are collectively referred to as the **bus**. (A **protocol** is a set of rules and standards that any two entities use for communication.) The paths, or lines, of the bus that are used to carry data are the part of the bus that we are most familiar with. Binary digits (0s and 1s) travel down this data path side by side. Some buses have data paths that are 8, 16, 32, or 64 bits wide. For example, a bus that has 32 wires, or lines, that data can travel on is called a 32-bit bus. A system board can have more than one bus, each using a different protocol, speed, data path size, and so on.

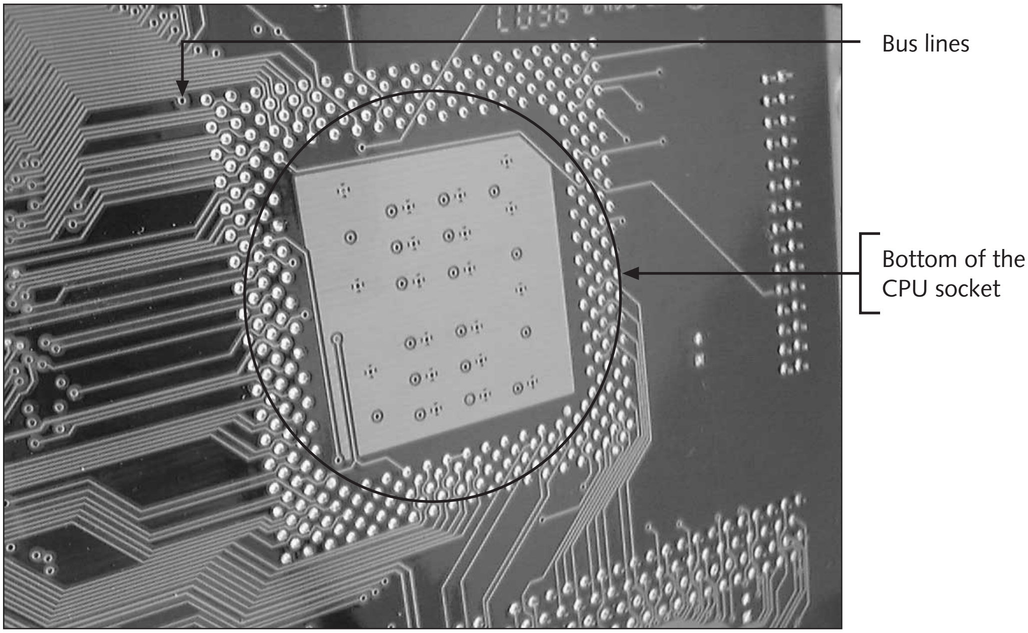

Figure 1-15 On the bottom of the system board, you can see bus lines terminating at the CPU socket

One of the most interesting lines, or circuits, on a bus is called the **system clock** and is dedicated to timing the activities of the chips on the system board. A crystal on the system board (Figure 1-16), similar to that found in watches, generates continuous pulses to produce the system clock. These pulses are carried on wires over the system board to chips and expansion slots to ensure that all activities are performed in a synchronized fashion by providing a beat, much as a metronome keeps the beat for a musician. If you can imagine the system clock as the conductor maintaining the rhythm of a symphony, then you can imagine hundreds of symphonies occurring within the computer every second as all devices that make up your computer function in lock-step at the right time at an incredibly fast beat!

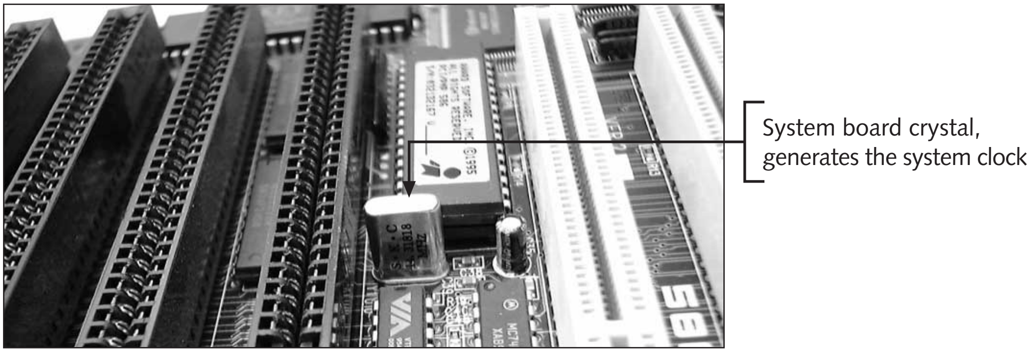

Figure 1-16 The system clock is a pulsating electrical signal sent out by this component that works much like a crystal in a wristwatch. One line, or circuit, on the system board bus is dedicated to carrying this pulse.

The lines of a bus often terminate at expansion slots (Figure 1-17). The size and shape of an expansion slot are dependent on the kind of bus it is using. Therefore, one way to determine the kind of bus you have is to examine the expansion slots on the system board. In Figure 1-8, there are three types of expansion slots showing. With a little practice, you can identify these slots by their length, by the position of the breaks in the slots, and by how far from the edge of the system board a slot is positioned. Look for all three of these expansion slots in Figure 1-8:

- PCI (peripheral component interconnect) expansion slot used for high-speed input/output devices
- AGP (accelerated graphics port) expansion slot used for a video card
- ISA (Industry Standard Architecture) expansion slot used by older and/or slower devices

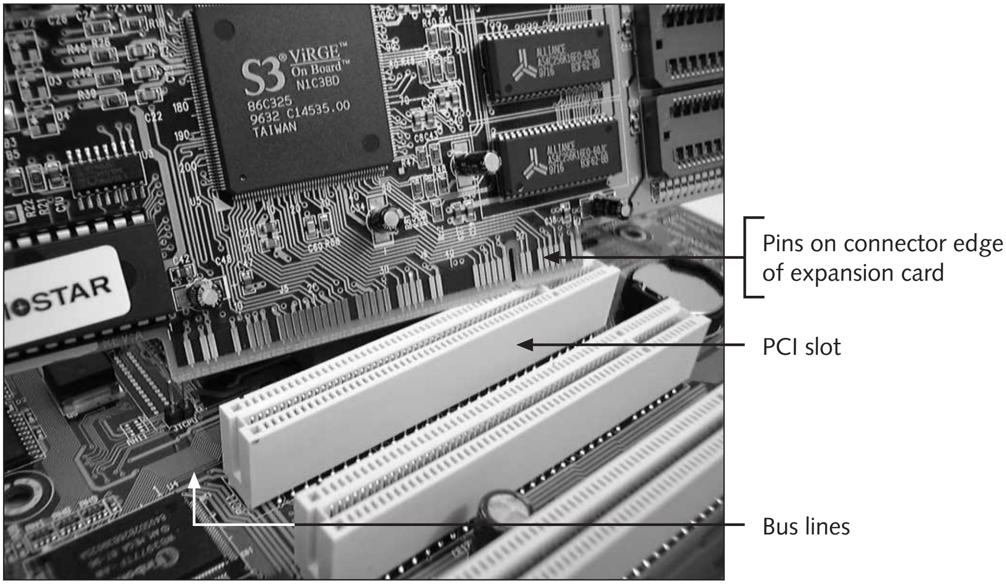

Figure 1-17 The lines of a bus terminate at an expansion slot where they connect to pins that connect to lines on the expansion card inserted in the slot

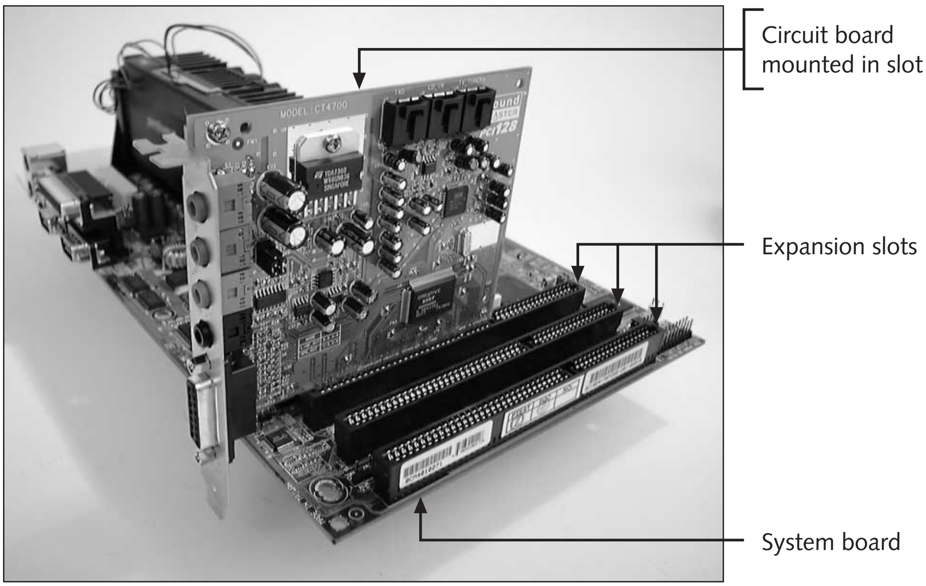

Figure 1-18 Circuit boards are mounted in expansion slots on the system board

Interface (Expansion) Cards

A+CORE 1.2

Circuit boards other than the system board inside the computer are sometimes called circuit cards, adapter boards, expansion cards, interface cards, or simply **cards** and are mounted in expansion slots on the system board (see Figure 1-18). Some common circuit boards you might find plugged into the expansion slots on your computer's system board are a card to control video, called a **video card** (see Figure 1-19), a sound card and a network card. These cards all enable the CPU to connect to an external device or, in the case of the network card, to a network. If your computer has an internal modem, that modem is installed as an expansion card. The modem technology is embedded on the card itself, and the card provides a port for a phone-line connection.

The easiest way to determine the function of a particular expansion card (short of seeing its name written on the card, which doesn't happen very often) is to look at the end of the card that fits against the back of the computer case. The card provides access to the outside of the case through a connector or port attached to the card (see Figure 1-19). A network card, for example, has a port designed to fit the network cable. An internal modem has one, or usually two, telephone jacks as its ports. A standard I/O controller card usually has at least one parallel port designed to fit a printer cable connection and a serial port designed to fit a modem cable connection or another serial device. (Parallel and serial refer to the way data and instructions are transmitted along a cable. With parallel transmission, streams of bits flow parallel to each other, while with serial transmission, single-file bits flow in one long stream. This is explained further in Chapter 5.)

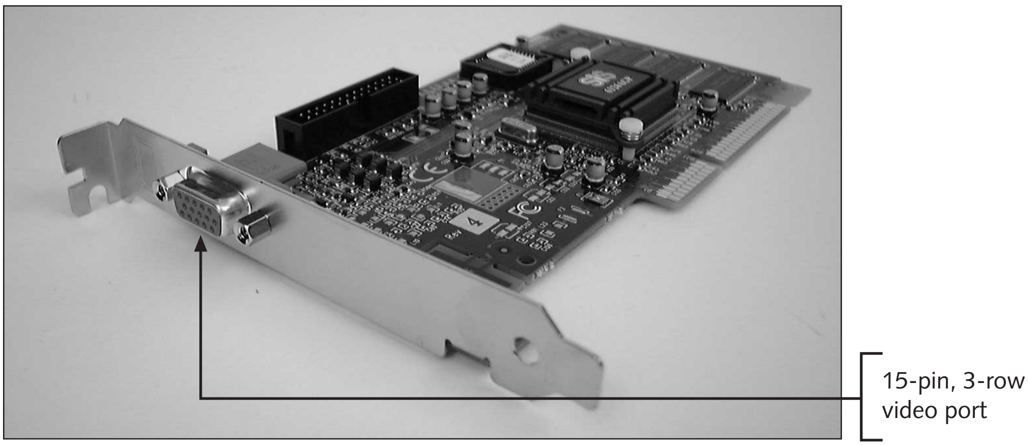

Figure 1-19 The easiest way to identify this video card is to look at the port on the end of the card

The Electrical System

A+CORE 1.1

The most important component of the computer's electrical system is the power supply, usually located near the rear of the case (see Figure 1-20). This **power supply** does not actually generate electricity but converts it and reduces it to a voltage that the computer can handle. Older power supplies had power cables that provided either 5 or 12 volts of DC current. Newer power supplies provide 3.3, 5, and 12 volts of DC current. In addition to providing power for the computer, the power supply runs a conventional fan directly from the electrical output voltage to help cool the inside of the computer case. When a computer is running, this fan and the spinning of the hard drive are the two primary noise makers.

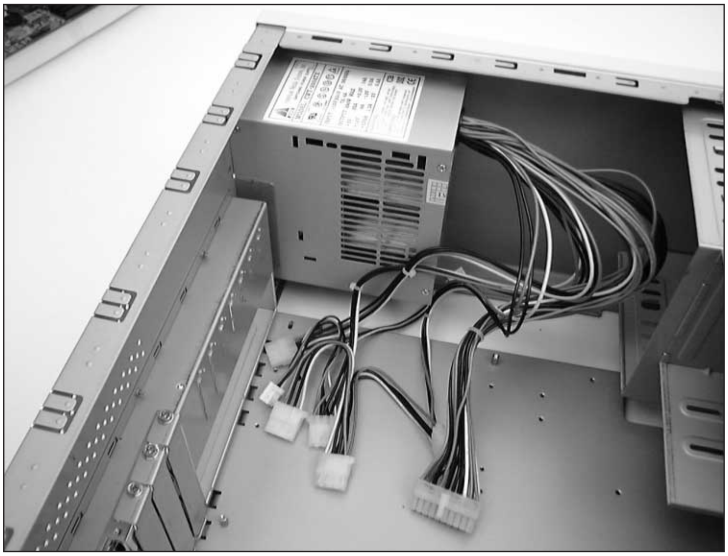

Figure 1-20 Power supply with connections

Every system board has one or a pair of connections to provide electrical power from the power supply to the system board and to other components that receive their power from ports and expansion slots coming off the system board (see Figure 1-21). Electricity travels over the system board on the buses. Each bus has wires, or lines, designated to carry voltage. Figure 1-22 shows the lines on the system board that are part of the bus ending at an expansion slot. Later, in Chapter 3, you will learn the specifics of which line carries what.

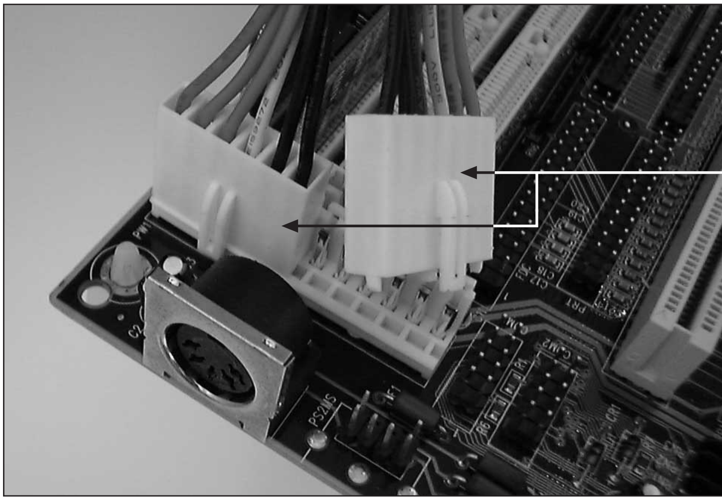

Figure 1-21 The system board receives its power from the power supply by way of one or two connections located near the edge of the board

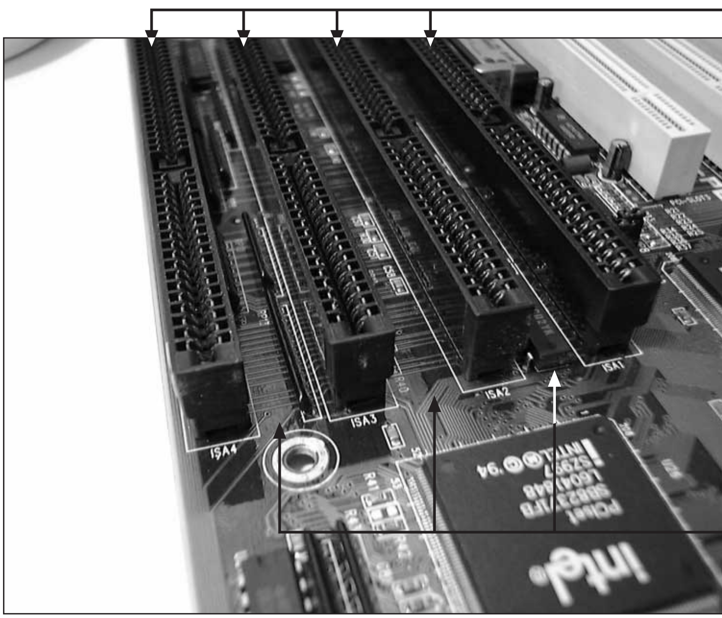

Figure 1-22 Bus lines ending at expansion slots

Instructions and Data Stored on the System Board

Some very basic instructions and data are stored on the system board apart from secondary storage, just enough to provide rudimentary information about the setup of the computer and to start the computer and look for software (stored either on a floppy disk or hard drive).

These data and instructions are stored on special chips on the system board (called ROM, or read-only memory, chips). Some data is also retained on the system board by way of setting physical switches on the board. In the case of these special chips, the distinction between hardware and software becomes vague.

A+ CORE 1.1, 1.8

Most of the time, it's easy to distinguish between hardware and software. For example, a floppy disk is hardware, but a file on the disk containing a set of instructions is software. This software file, sometimes called a **program**, might be stored on the disk today, but you can erase that file tomorrow and write a new one to the disk. In this case, it is clear that a floppy disk is a permanent physical entity, whereas the program is not. Sometimes, however, hardware and software are not so easy to distinguish. For instance, a ROM (read-only memory) microchip on a circuit board inside your computer has software instructions permanently etched into it during fabrication. This software is actually a part of the hardware and can never be erased. In this case, hardware and software are married together, and it's difficult to separate the two, either physically or logically. Some even give a new name to such hybrid components, calling them **firmware**.

These ROM chips hold programs or software that, among other things, tell the CPU how to perform many fundamental input/output tasks that manage the computer, and they are therefore sometimes called **BIOS** (**basic input/output system**) chips. See Figure 1-23 for an example of a **ROM BIOS** chip on a video expansion card. The system board contains a vital ROM BIOS chip (Figure 1-24) that contains the programming necessary to start the computer, and other fundamental BIOS programming for functions such as interacting with the floppy disk drive. If you want to upgrade these utility programs, usually you must buy new ROM chips. However, there are now ROM chips on the market that actually can be reprogrammed. Called **Flash ROM**, the software stored on these chips can be overwritten by new software that remains on the chip until it is overwritten.

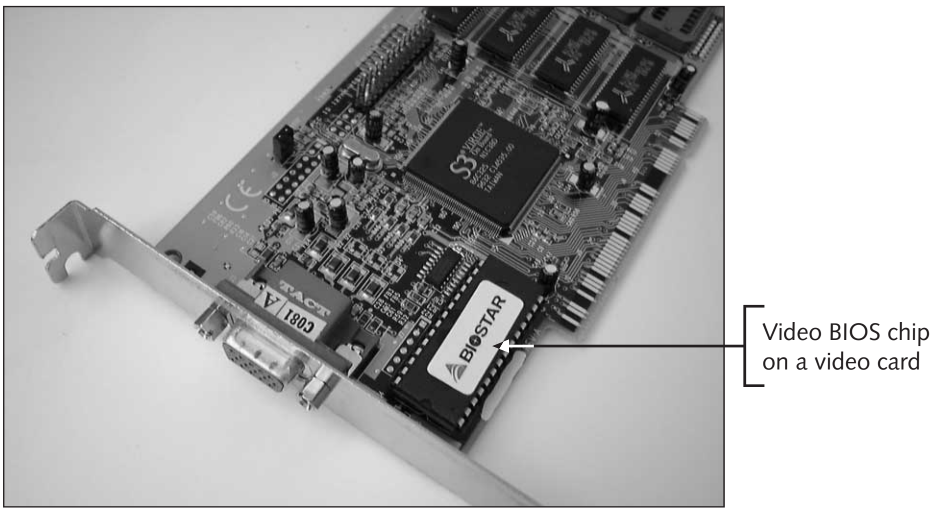

Figure 1-23 A ROM BIOS chip on a video card holds programs that provide instructions to operate the card

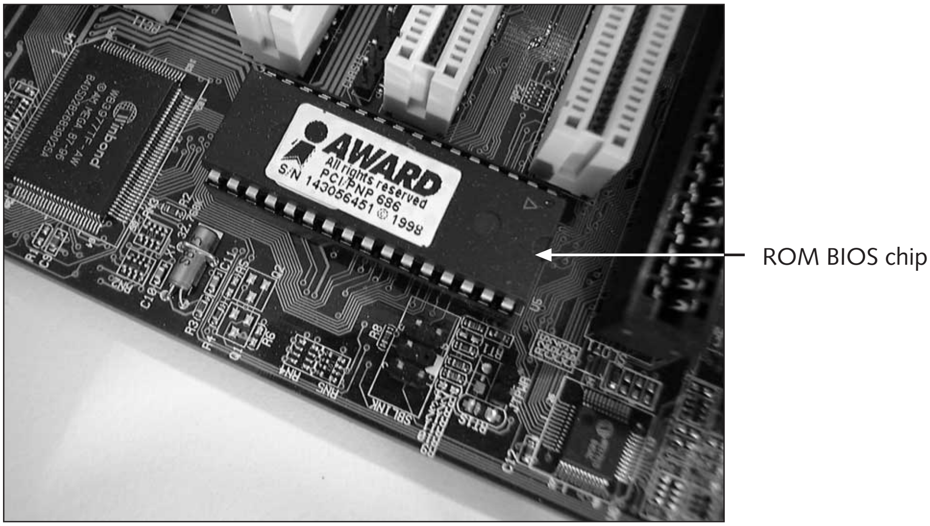

Figure 1-24 The ROM BIOS chip on the system board contains the programming to start up the PC as well as perform many other fundamental tasks

Another chip on the system board contains a very small amount of memory, or RAM, enough to hold configuration or setup information about the computer. This is called the **CMOS configuration chip**. This chip, shown in Figure 1-25, is responsible for remembering how memory is allocated, which hard drives and floppy drives are present, and so forth. When the computer is first turned on, it looks to this CMOS chip to find out what hardware it should expect to find. A similar function is performed on Macintosh computers by a chip called the parameter RAM (PRAM) chip. The CMOS chip is powered by a trickle of electricity from a small battery—located on the system board or computer case, usually close to the CMOS chip itself—so that, when the computer is turned off, the CMOS chip still retains its data.

Even though a computer has many CMOS chips, in the computer industry, the term "CMOS chip" has come to mean the one chip on the system board that holds the configuration or setup information. If you hear the following: "What does CMOS say?" or "Let's change CMOS," the person is talking about the configuration or setup information stored on this one CMOS chip. The program to change CMOS setup is stored in the ROM BIOS chip and can be accessed during the startup process.

A system board can also retain setup or installation information in different settings of jumpers or DIP (dual in-line package) switches on the board. **Jumpers** are considered open or closed based on whether a jumper cover is present on two small wires that stick up off the system board (see Figure 1-26). A **DIP switch** is similar to a light switch and is on or off based on the direction the small switch is set. Most system boards have at least one, often several, jumpers, and perhaps a single bank of DIP switches, although the trend is to include most setup information in CMOS rather than to have a jumper or switch on the board that has to be mechanically set.

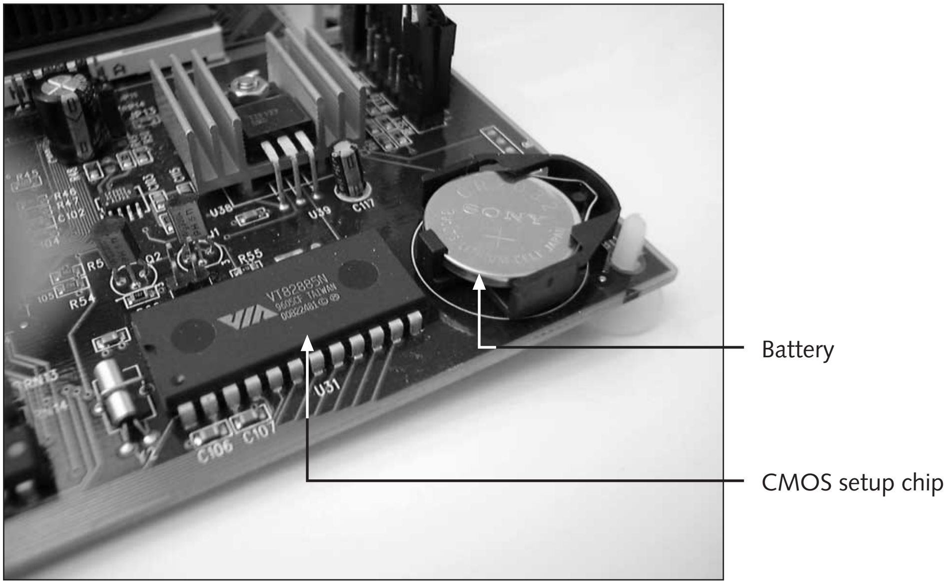

Figure 1-25 The CMOS setup chip, powered by a battery when the PC is turned off, contains data about the system configuration as well as the current time and date

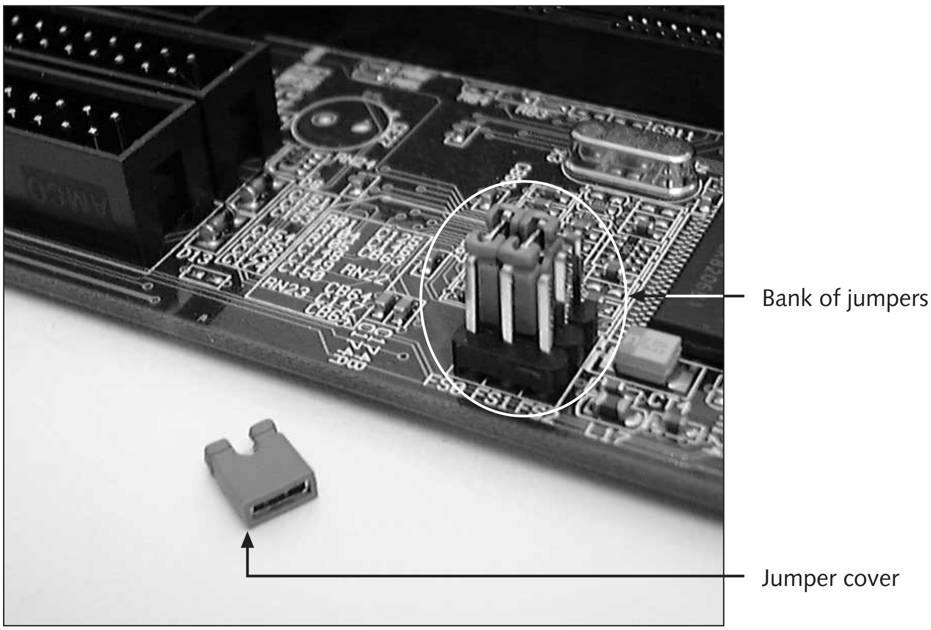

Figure 1-26 Setup information about the system board can be stored by setting a jumper on (closed) or off (open). A jumper is closed if the cover is in place, connecting the two wires that make up the jumper; a jumper is open if the cover is not in place.

SOFTWARE

Let's return to the analogy used earlier (Figure 1-1) when we compared a car to computer hardware. If a car is like a hardware system, then the people who service and drive a car are like software that manages and instructs the hardware. The mechanic and driver make the automobile a functioning tool that can be used to accomplish a task. Without their intelligence, skill, and direction, the car is nothing more than an interconnected assemblage of electronic and mechanical devices. Software provides a similar function for hardware. Software is the intelligence of the computer; it determines what hardware is present, decides how it is configured and used, and then uses that hardware to perform tasks.

Three Types of Software and What They Do

Software consists of programs written by programmers that instruct computers to perform specific tasks. Almost all PC software falls into three categories: firmware (BIOS), operating system (OS), and applications software. The BIOS and OS perform tasks at startup that determine the overall health and functionality of the computer, just as a mechanic is responsible for the condition of a car before it is turned over to the driver. After startup, the OS, working with applications software and BIOS, provides instructions to the hardware to perform tasks, much as a driver operates the automobile.

As seen in Figure 1-27, first BIOS and then the OS is in control of preparing the computer for user interaction when it is first turned on. The user can then interact directly with the OS to perform simple tasks such as copying files from the hard drive to a disk or installing applications software. Or the user can use applications software to perform higher-level tasks such as word processing or database management. In this case, the OS is still working behind the scenes, serving as a middle layer between the applications software and the hardware.

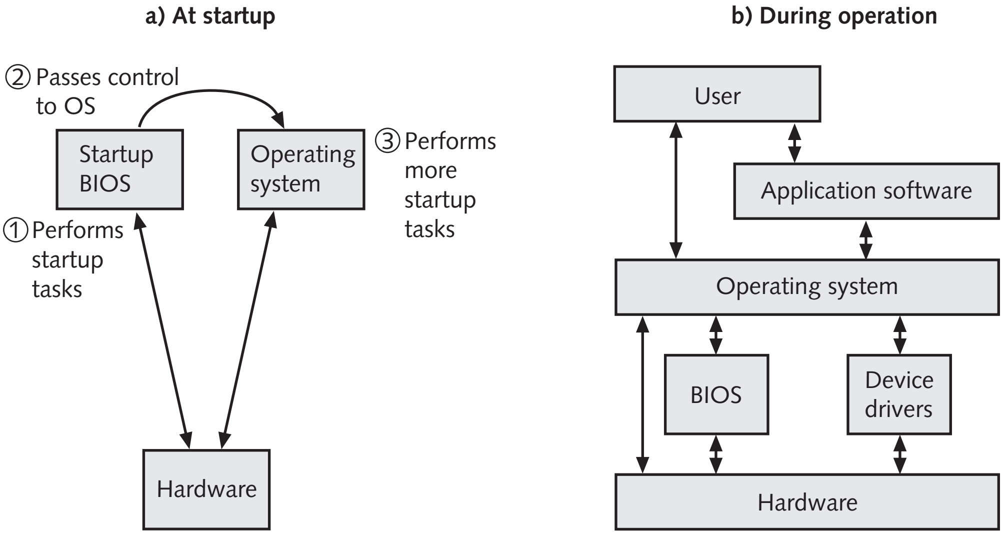

Figure 1-27 Not all software relates directly to hardware; some is dependent on other software to perform many basic functions

A+CORE 1.1

Firmware or BIOS

BIOS on the system board and other circuit boards provides some fundamental instructions to hardware and often serves as the interface between higher-level software and hardware. The BIOS programs stored on the system board are together referred to as **system BIOS**, or **on-board BIOS**. Part of system BIOS is **startup BIOS**, which runs many computer startup functions and brings the computer to a state in which it can be managed by the OS. Once startup BIOS has completed its tasks, it turns control over to the OS, which stays in control as long as the PC is turned on. The process of powering up and bringing the PC to a ready state is called **booting** and is discussed in detail in Chapter 2.

Software Layers

Once the computer is running, a hierarchy determines how software interacts so that high-level software can depend on low-level software to manage the hardware for it. Looking again at Figure 1-27, you can see that applications software depends on the OS to interface with hardware. The OS might instruct the hardware directly, use BIOS to provide the instructions, or use special software designed to interface with specific hardware devices. These special software programs are called **device drivers** and serve the same functions as BIOS programs, but they are stored on secondary storage devices such as a hard drive rather than on ROM chips, as BIOS is. Some device drivers are provided by the OS, and some are provided by the manufacturer of the specific hardware device with which they are designed to interface.

TIP

You can find device drivers in a number of sources. Some come with the operating system. And the manufacturer provides some either with the device when it is purchased or over the Internet.

One advantage of using BIOS and device drivers to interface with hardware is that it frees the OS or applications software from knowing the specifics of how to communicate with the device. For example, different printers understand data and commands according to different sets of rules and standards called protocols. Applications software and the OS can pass print requests to the printer driver, which communicates with the printer, as in Figure 1-28. With the device drivers doing the interpreting, applications software developers do not have to include the specific protocol and standards for every printer that might be used by the applications they write.

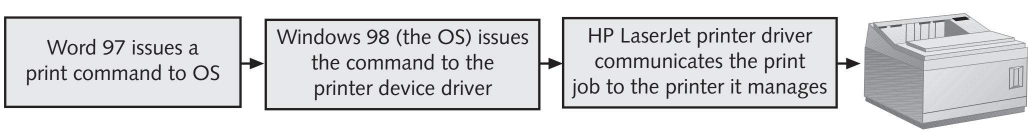

Figure 1-28 Any specific protocols and commands sent to the printer are the responsibility of the printer device driver

The applications software does not even need to know which printer is being used, because Windows keeps track of the **default printer**, the currently selected printer. The application sends print jobs to Windows for printing. Windows uses the default printer unless the user selects a different one from the Windows printer list. Windows knows which device driver to call to execute the print job because the device driver was assigned to that printer when the printer was installed.

How Software Manages and Shares Information

Before the CPU can process data or follow software instructions, the data or instructions must first be stored in RAM. The CPU tracks where this information is stored in RAM by assigning an address to each unit of RAM that can hold one byte of information. These addresses are called **memory addresses** and are most often displayed on the screen as hexadecimal (base 16) numbers in segment/offset form (for example, C800:5). See Appendix D for more information on the hexadecimal number system as it applies to memory addresses.

As you saw in Figure 1-27, BIOS, device drivers, the OS, and applications software are working when a computer is running. During output operations, applications software must pass information to the OS, which in turn passes that information to a device driver or to BIOS. BIOS and device drivers managing input devices must pass information to the OS, which passes it to the applications software.

Using the CPU, software processes and shares data by referring to the memory address of the data, as seen in Figure 1-29. For example, if applications software wants to print data, rather than saying to the OS, "Please print data ABC," it says to the OS, "Please print the data at memory address 123." The OS then turns to the printer device driver and says, "Please send data at memory address 123 to the printer you control."

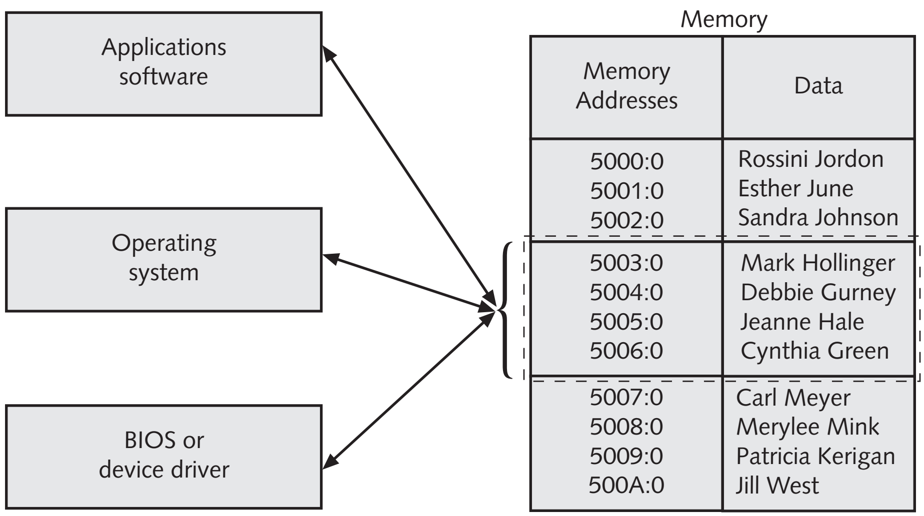

Figure 1-29 Software can exchange information by storing data in RAM that has been assigned memory addresses. Memory addresses can then be communicated to other software layers.

Operating Systems

Although there are several operating systems on the market, a microcomputer needs only one to operate. Different OSs support different types of hardware systems and user needs. The first OS among IBM computers and IBM-compatible computers was DOS, the disk OS. Because DOS was written for early PCs, it still has significant limitations. For years, DOS was used with Windows, a user-friendly intermediate program between DOS and the user. Versions of Windows include Windows 3.1 and Windows 3.11 (also called Windows for Workgroups), which together are referred to as Windows 3.x. With Windows 3.x, DOS is the OS. The more recent Windows 95 and Windows 98, which we call Windows 9x, provide a user-friendly interface and actually conduct OS functions. Other OSs include Macintosh for Apple computers, UNIX, OS/2, and Windows NT. UNIX and Windows NT are popular OSs used for high-end PCs on networks and to support applications used on the Internet. This section looks at different OSs, paying careful attention to their similarities and differences.

In general, OSs have evolved as hardware has improved. Software tends to run behind hardware in development, because software is generally designed to use current hardware, rather than vice versa. As we study the different OSs in this chapter, take note of that evolution process. Operating systems serve many functions. Among them are the following:

- Managing BIOS
- Managing files on secondary storage devices
- Managing primary memory (RAM)
- Diagnosing problems with software and hardware
- Interfacing between hardware and software (that is, interpreting applications software needs to the hardware and interpreting hardware needs to applications software)
- Performing housekeeping procedures requested by the user, often concerning secondary storage devices, such as formatting new disks, deleting files, copying files, and changing the system date

While DOS has served computing well, it has serious computing limitations, primarily because it has been modified extensively as hardware capabilities have increased. When computer users started using DOS in the early 1980s, computer systems were very small, and the power and usefulness of the software and hardware were limited. The computer industry made assumptions and decisions based on limitations in hardware complexity and amount of memory that still limit us today, even though the complexity and memory have increased astronomically. Such outdated limitations particularly affect the way hardware and software interact through the OS. Some drawbacks of DOS were partly solved by Windows 3.1 and Windows 3.11. Windows 95 and Windows 98, which contain parts of DOS and parts of a completely rewritten OS, offer even greater improvements.

Starting Up the Operating System

Operating systems are stored on hard drives, but applications software can be stored on hard drives, floppy disks, or CD-ROMs. Software is stored as files designated as **program files**.

A **file** is a collection of data or programmed instructions that is stored on a secondary storage device and assigned a single name, which the OS uses to identify it.

Although the OS is stored in files on the hard drive, these stored programs cannot be executed from their secondary storage locations. As explained above, these instructions first must be copied from secondary storage into RAM, or memory. The CPU can then read from one memory location in RAM to another to receive and follow instructions.

> TIP Remember that *all* programs must be copied into RAM before the CPU can read them. This includes even the operating system, which resides permanently on secondary storage (usually a hard drive) and must be copied or loaded into RAM each time the computer is started.

Figure 1-30 shows this two-step process. Recall that startup BIOS is in control when the computer is first turned on. When startup BIOS is ready to load the OS, it finds the OS program file stored on the hard drive and copies the file into RAM. The program file contains a list of instructions; each instruction is stored in a separate memory location with a memory address assigned to it. In the second step, control is given to these instructions in RAM. Beginning with the first memory address, the CPU executes the instruction stored at that address. Sometimes, an instruction causes control to be sent to a memory address other than the next one in sequential order. This action is called a **program jump**, as demonstrated in the move from Instruction 4 to Instruction 6 in Figure 1-30.

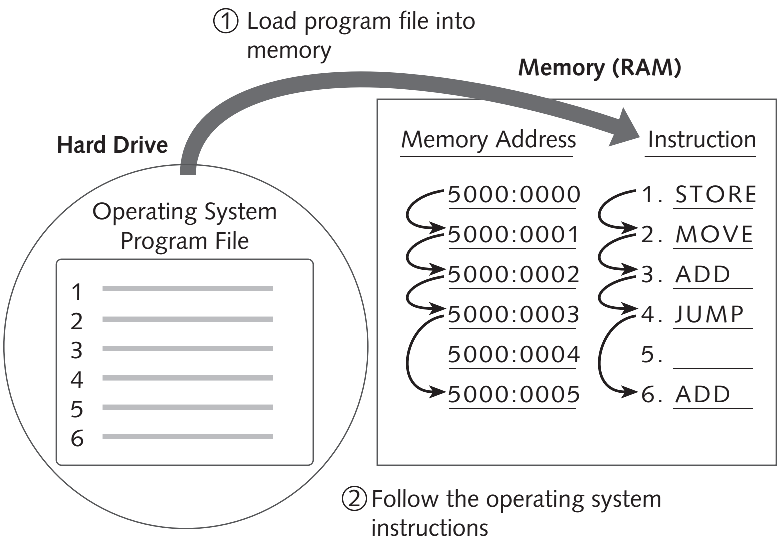

Figure 1-30 The operating system is stored in files on the hard drive but is executed from memory

Once the first OS program is in control, it will look for other OS program files on the hard drive and load them into memory. The OS consists of a group of programs, several of which

are stored in RAM at any one time. Once the OS is loaded into memory and given control, as part of the booting process, it will perform its own limited startup routines.

Interfacing with the Operating System

After booting is complete, the OS either automatically executes an applications software program or turns to the user for its next instruction. If you are working with the OS, you will see an interface on the monitor screen. This interface can be one of three types:

A Command-driven Interface With a command-driven interface, you key in command lines to tell the OS to perform operations (see Figure 1-31). For instance, the command-driven interface of DOS is the C prompt, which looks like this:

```
C> or C:\>
```

Computer users who are good typists and are very familiar with DOS commands often prefer this kind of OS interface.

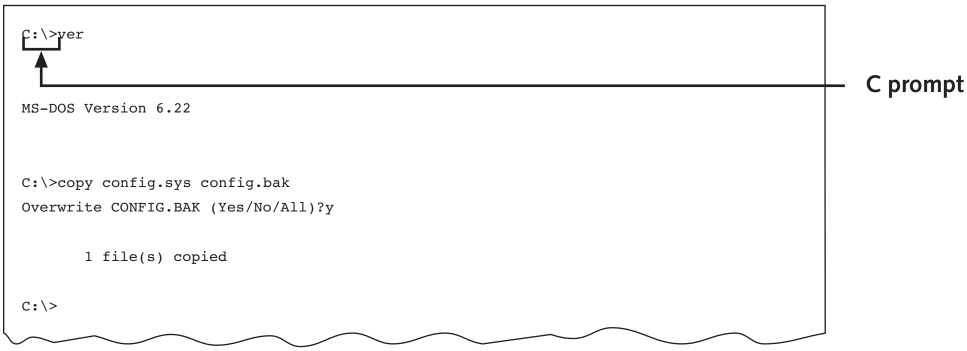

Figure 1-31 An operating system command-driven interface: the C prompt

A Menu-driven Interface Some OSs allow you to choose from a list of options displayed on the screen. An example of such a menu-driven interface is Explorer in Windows 98. From the drop-down menus, you can format disks, rename files, copy and delete files, and perform many other operations to manage files and storage devices (see Figure 1-32).

An Icon-driven Interface Today's OSs are more likely to use an icon-driven interface than a command-driven one. With an icon-driven interface, sometimes called a **graphical user interface** or **GUI**, you perform operations by selecting icons (or pictures) on the screen. When an OS is first executed, the initial screen that appears, together with its menus, commands, and icons, is called the **desktop**. Figure 1-33 shows the Windows 98 default desktop, which has an icon-driven interface. You double-click an icon with the left mouse button to execute an applications software program. Just about all OSs today offer icon-driven interfaces.

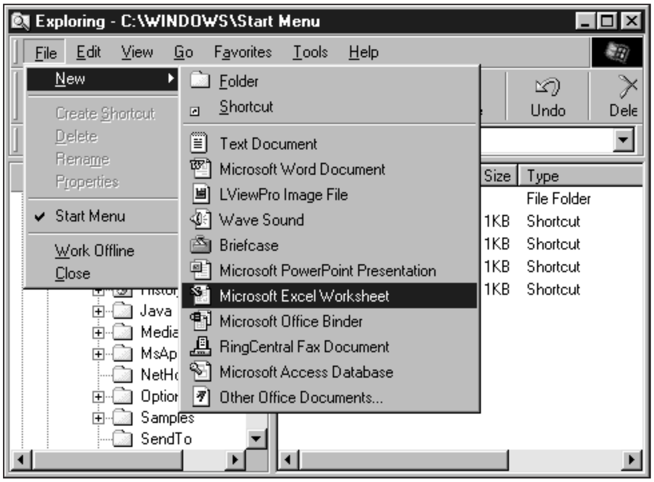

Figure 1-32 A menu-driven interface: Explorer in Windows 98

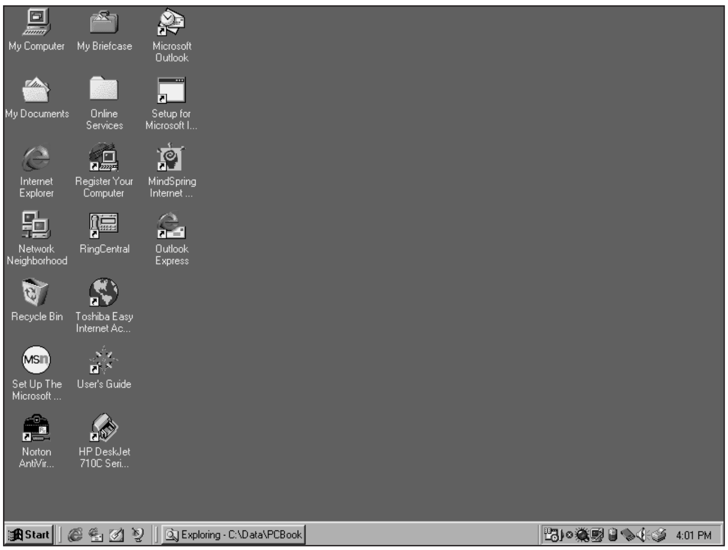

Figure 1-33 An icon-driven interface: Windows 98 desktop

Survey of Operating Systems

Operating systems continue to evolve as hardware and software technologies improve. As you look at several OSs, you will see the evolution process from DOS to DOS with Windows 3.x, to Windows 9x. To understand these gradual improvements in OSs, you need to understand the following terms:

- Multitasking. **Multitasking**, as it applies to hardware, refers to the ability of a CPU to do more than one thing at a time. The first CPU for microcomputers with this ability was the Pentium by Intel. The older i386 and i486 CPUs could

do only one thing at a time (80386 and 80486 CPUs are often abbreviated as i386 and i486 in documentation; the i stands for the chip manufacturer, Intel). Earlier OSs did not need to support multitasking; newer OSs now need to support some form of multitasking.

- Cooperative multitasking. **Cooperative multitasking**, sometimes called **task switching**, is not true multitasking, in that the CPU is only doing one thing at a time. The CPU is switching back and forth between applications so that more than one application can be loaded at the same time. There is only one active application and one or more inactive applications sitting in the background waiting for the active application to relinquish control.

You observe cooperative multitasking when you have two applications open, each in its own window. You don't need to close one application before opening another. DOS does not handle cooperative multitasking, but Windows does (see Figure 1-34).

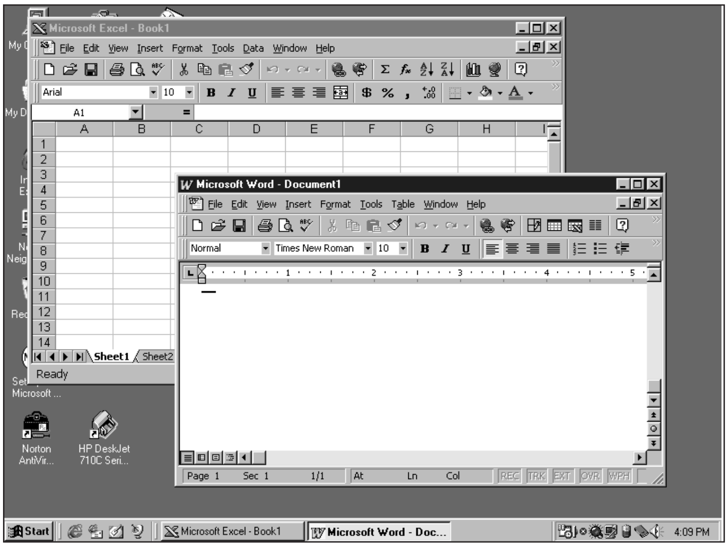

Figure 1-34 A multitasking environment allows two or more applications to run simultaneously

- Preemptive multitasking. **Preemptive multitasking** is another type of pseudo-multitasking whereby the OS allots CPU time to an application for a specified period, and then preempts the processing to give the CPU to another application. The end result is that the computer appears to be doing true multitasking.

- Environment. **Environment** refers to the type of support the OS provides to the applications software. For example, in order for applications software to offer you a window with mouse movement, buttons to click, and icons to view, it must be supported by a GUI environment, such as Windows. Such an application is said to need a "GUI environment" to work. Another example is the DOS environment

that offers to its applications software only a "single-tasking environment." The software does not expect another applications software package to be running concurrently with it.

- Windows 3.x operating environment. There is a unique situation with Windows 3.x. It's not really an OS, but neither does it act like normal applications software. Windows 3.x provides an operating environment, which refers to the overall support that it provides to applications software, including cooperative multitasking and a GUI (which DOS does not offer). Windows 3.x is the "middleman" that manages this pseudomultitasking environment by passing tasks to DOS one at a time. DOS manages its single-tasking environment and relates to the hardware in single-task fashion. Windows provides the cooperative multitasking environment to the applications.

However, as Figure 1-35 shows, Windows 3.x both performs some of the functions of an OS and provides the environment within which applications software works. Figure 1-35 shows that the applications software relates only to Windows. There is usually no attempt to make applications software interact directly with DOS, system BIOS, or the hardware in a Windows environment. Windows 3.x sometimes interfaces directly with hardware, such as when printing with one of its own printer drivers. In this case, Windows 3.x is serving as the OS. Sometimes, however, Windows passes such functions to DOS. DOS can choose to relate directly to the hardware or it can choose to pass the hardware request to BIOS or a device driver to be processed.

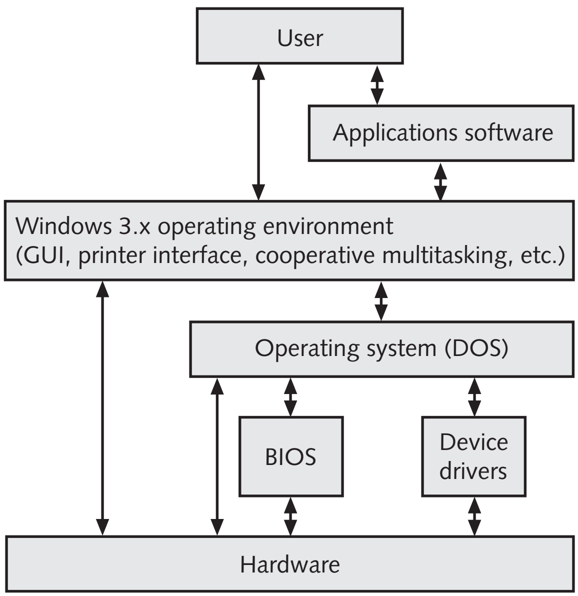

Figure 1-35 Windows 3.x is unique in that it is an extra software layer between the OS and applications software. Compare this figure to Figure 1-27b.

With the above discussion of operating system characteristics in mind, we will compare several OSs based on these criteria:

- What kind of interface does the OS provide for the user?
- Can the OS support some form of multitasking?
- Can the OS easily manage large quantities of primary memory (RAM) and secondary storage?
- How many and what kinds of applications are written to work with the OS?
- How powerful must the hardware be to make efficient use of the OS?
- How does the OS perform in a network?

Tables 1-1 through 1-8 list the advantages and disadvantages of the most well-known OSs used in PCs and the OS used in Macintoshes. When choosing an OS, consider all the criteria discussed in these tables. Your choice will be determined by the size and type of your microcomputer system, your familiarity with the various OSs, and the applications software you plan to use.

DOS (Disk Operating System) The first OS used by IBM microcomputers; for years DOS remained the unchallenged standard for OSs used by IBM and IBM-compatible machines. Most seasoned microcomputer users are comfortable and familiar with DOS. Table 1-1 summarizes its advantages and disadvantages.

Table 1-1 Advantages and disadvantages of DOS

Advantages	Disadvantages
• DOS runs on small, inexpensive microcomputers with a minimum amount of memory and hard drive space. • Some older applications are still in use today that were written for DOS and older hardware because of the low overhead of DOS compared to more modern OSs. • DOS is still a viable option for some specialized applications using a dedicated computer that does not involve heavy user interaction —for example, a microcomputer dedicated to controlling an in-house phone system. • DOS can be used to boot up and troubleshoot a computer when a more sophisticated OS is too cumbersome and has too much overhead.	• Memory management is awkward and sometimes slow. • DOS has no icon-driven interface. • DOS does only single-tasking; that is, it supports only one application running at a time. • DOS was not designed for use on networks. A separate applications software program is necessary for a DOS machine to access a network. • The last standalone version is DOS 6.22, which does not take advantage of the many new CPU features now available. (However, Windows 9x has a newer DOS core.) • Hardly any new software is being written for DOS.

DOS with Windows 3.1 and 3.11 Operating Environments Written by the same software company that owns DOS (Microsoft Corporation), Windows 3.x was designed to allow

applications to have a graphical interface, which is popular with users. DOS runs in the background as the true OS and uses Windows 3.x as a middle layer, or "go-between" program, between the application and DOS, providing the operating environment for applications software packages. When Windows is running, users can interact with applications directly with Windows or directly with DOS. A point of historical interest: Windows 3.11, called Windows for Workgroups, was the first Windows environment designed to interface with a network without depending on separate applications software to do the job. Table 1-2 summarizes the advantages and disadvantages of using DOS with Windows 3.x, which has been replaced with Windows 9x.

Table 1-2 Advantages and disadvantages of DOS with Windows 3.x

Advantages	Disadvantages
• DOS with Windows 3.x provides an icon-driven interface. • Windows supports cooperative multitasking as it manages more than one open application by passing segments to DOS, which then, in turn, interfaces with hardware. • DOS with Windows 3.x can run on relatively inexpensive microcomputer systems with present-day CPUs. • Many applications software programs are still in use that were written to run on Windows 3.x.	• Memory management is awkward and sometimes slow. • DOS with Windows 3.x is sometimes slow due to the complexity of the middle layer, or "go-between," concept. • The DOS/Windows 3.x combination does not take advantage of the full computing power of minimum amount of hard drive space. • There is no new software being written today specifically for Windows 3.x. • DOS and Windows applications tend to conflict with each other when sharing hardware devices.

Windows 95 and Windows 98 Windows 95 and Windows 98 (referred to as Windows 9x) take us two steps closer to a new OS but do not completely eliminate DOS. Windows 95 is the marriage of Windows for Workgroups (Windows 3.11) with an updated version of DOS sometimes known as DOS 7.0, together with some completely new additions and improvements to the OS. Windows 95 also introduced an improved and more automated method of installing new hardware devices, called **Plug and Play**. Windows 98 made several improvements on Windows 95 but is fundamentally the same OS. Windows 98 offers additional support for larger hard drives and more hardware devices, includes more software utilities, and is faster than Windows 95. The underlying DOS portion of Windows 98 is named DOS 7.1. Microsoft has announced that Windows 98 will be the last OS that it distributes that has a DOS foundation. Currently, there have been two editions of Windows 98 released. The third edition, code-named Windows Millennium Edition, is scheduled to be released before this book is in print. Table 1-3 summarizes the advantages and disadvantages of Windows 9x.

Table 1-3 Advantages and disadvantages of Windows 9x

Advantages	Disadvantages
• Windows 9x offers a very user-friendly and intuitive GUI interface. • Windows 9x offers almost complete backward compatibility for applications written for DOS and earlier versions of Windows. • Windows 9x is a mix of older and newer OS technology and allows both older and newer software and hardware to run. • Windows 9x offers the ability for one PC to talk with another over phone lines without additional software. It works well for low-end network use, such as when two users want to exchange files. • Disk access time under Windows 9x is improved over DOS and Windows 3.x. • Plug and Play features make installing some new hardware devices easier than with earlier OSs. • Windows 9x supports preemptive multitasking. While the hourglass is showing on the window of an application, you can make another application active by clicking on its window.	• Windows 9x requires at least a 386 CPU, 8 MB of RAM, and 30 MB of hard drive space, thus prohibiting its use on some older PCs. • Because of the attempt to bridge older and newer technology, there are some problems with failures and errors created in this hybrid environment.

UNIX Operating System UNIX originally was written for mainframe computers in the early 1970s; only in the past few years has it become available for many different kinds of computers, including PCs, and it is now a popular OS for networking. UNIX computers are often used for Internet support. Problems with UNIX stem mostly from the lack of consistency from one vendor's version to another.

A variation of UNIX that has recently gained in popularity is Linux (pronounced "LIH-nucks"), an OS originally created by Linus Torvalds when he was a student at the University of Helsinki in Finland. This OS is free to everyone, and all the underlying programming instructions (called source code) are also freely distributed. For more information on Linux, see *www.linux.org*. Linux is popular as a low-end, scaled-down version of UNIX that can be used to support small networks or as a training tool to learn UNIX. Table 1-4 summarizes the advantages and disadvantages of the UNIX OS.

Table 1-4 Advantages and disadvantages of UNIX

Advantages	Disadvantages
• UNIX was written for powerful microcomputer systems and has strong multitasking capability, including preemptive multitasking. • UNIX manages large quantities of memory well. • UNIX performs very well in a networking environment.	• UNIX industry standards are not uniform, making it difficult for UNIX developers, administrators, and users to move from one UNIX vendor to another. • UNIX requires a powerful, large microcomputer system. • Few business applications software packages have been written for UNIX for PCs, although there are several very powerful database packages available under UNIX, such as Informix and Oracle.

Windows NT Windows NT (New Technology) breaks with previous versions of Windows. Although older applications written for DOS and DOS with Windows might work under Windows NT, Windows NT developers do not guarantee compatibility between the OSs. It takes an aggressive and altogether new approach toward managing hardware resources and interfacing with applications software. Windows NT completely eliminates the underlying relationship with DOS.

TIP

Remember that Windows NT is the one Windows OS that does not guarantee compatibility with programs written for earlier Windows versions. That's because NT eliminates the connection to DOS. Whereas even Windows 98 contains some DOS in its programming code, NT does not. Older DOS and DOS with Windows applications *might* work with Windows NT, but you can't count on it.

Windows NT supports preemptive multitasking and also supports a system that contains two or more CPUs, called multiprocessing. Multiprocessing provides true multitasking because the OS can manage two or more processes happening at the same time, each using its own CPU.

Windows NT is designed to work within a powerful networked environment. Computers (called **servers**) are configured to store programs and data used remotely by other computers (called **clients**). With client-server arrangements, an organization's resources can be used more effectively, since computers are networked together to share these resources. Windows NT Workstation is an OS designed to run on the client, and Windows NT Server runs on the server. Table 1-5 summarizes the advantages and disadvantages of Windows NT.

Table 1-5 Advantages and disadvantages of Windows NT

Advantages	Disadvantages
• Windows NT is designed to run in powerful client-server environments and targets both the client and the server market. • Windows NT offers a completely new file management system, different from earlier Windows OSs. • Windows NT Workstation offers both networking over a LAN and dial-up connections over phone lines. • Windows NT Server offers powerful security both as a file server and for network administration. • Windows NT supports preemptive multitasking and multiprocessing.	• Windows NT requires at least a 486 CPU, 16 MB of RAM, and 120 MB of hard drive space, thus eliminating it as a plausible option for older, low-end PCs. • Windows NT is not compatible with some older hardware and software.

Windows 2000 Windows 2000 is actually a suite of operating systems, each designed for a different sized computer system. Windows 2000 is built on the Windows NT architecture and is designed to ultimately replace both Windows 9x for low-end systems and Windows NT for midrange and high-end systems. Windows 2000 includes four operating systems:

- **Windows 2000 Professional** This OS is designed to ultimately replace both Windows 9x and Windows NT Workstation as a personal computer desktop or notebook OS. It is an improved version of Windows NT Workstation, using the same new technology approach to hardware and software, and includes all the popular features of Windows 9x, including Plug and Play.
- **Windows 2000 Server** This OS is the improved version of Windows NT Server and is designed as a network operating system for low-end servers.
- **Windows 2000 Advanced Server** This network operating system has the same features as Windows 2000 Server, but is designed to run on more powerful servers. It supports up to eight processors on one machine, and up to 8 GB (gigabytes) of memory.
- **Windows 2000 Datacenter Server** This network operating system is another step up from Windows 2000 Advanced Server and is designed to support up to 32 processors and up to 64 GB of memory. It is intended to be used in large enterprise operations centers.

Table 1-6 lists the advantages and disadvantages of Windows 2000.

Table 1-6 Advantages and disadvantages of Windows 2000

Advantages	Disadvantages
• Windows 2000 provides powerful support to a network, including advanced security for the network and the ability to organize access to network resources in a centralized location on the network called an Active Directory. • Windows 2000 is backward compatible with all Windows NT and Windows 9x applications and most Windows 3.x and DOS applications. • Windows 2000 is really four operating systems, each targeting a different sized computer and different computing needs, therefore making the OS suite extremely versatile.	• Just as with Windows NT, Windows 2000 hardware requirements disqualify it as an option for an older, low-end PC operating system. Minimum requirements for Windows 2000 Professional are a 133 MHz Pentium-compatible CPU, 64 MB of RAM, and 650 MB of hard drive storage. • Windows 2000 is not scalable. Rather than having one OS that can easily handle a major computer system upgrade, the user must purchase one version of Windows 2000 for a small system and another to handle the upgraded system. • Windows 2000 is not as stable as other operating systems used to manage a large network.

OS/2 OS/2, written by IBM in cooperation with Microsoft Corporation, provides an altogether different OS in place of DOS. Errors in earlier versions and large computer hardware requirements have made OS/2 slow to gain popularity. Table 1-7 summarizes the advantages and disadvantages of OS/2.

Table 1-7 Advantages and disadvantages of OS/2

Advantages	Disadvantages
• OS/2 supports preemptive multitasking. • OS/2 can handle large quantities of memory directly and quickly. • OS/2 has an icon-driven interface. • OS/2 works well in a networking environment.	• Relatively few applications software packages are written for OS/2. Some consider it a dead or dying OS, although it is still used by some. • Many microcomputer users are not familiar with OS/2 and avoid it for that reason. • OS/2 requires a powerful computer system and large amounts of RAM and hard drive space to run efficiently.

Macintosh Operating System Available only on Macintosh computers, several versions of the Macintosh OS have been written, the latest being Mac OS X (ten), which offers easy access to the Internet and allows any Macintosh computer to become a Web server for a small network. Table 1-8 summarizes the advantages and disadvantages of the Macintosh OS.

Table 1-8 Advantages and disadvantages of the Macintosh operating system

Advantages	Disadvantages
• The Mac OS has an excellent icon-driven interface, and it is easy to learn and use. • The Mac OS supports cooperative multitasking. • The Mac OS manages large quantities of memory. • Many applications exist for the Mac OS to create and edit graphics, build web sites, and manage multimedia.	• Historically, the Macintosh was not viewed as a professional computer but rather was relegated to education and game playing. Then the Mac gained a significant place in the professional desktop publishing and graphics markets. Most recently, the availability of more powerful IBM-compatible PCs and OSs to handle the high demands of graphics has reduced the demand for the Mac.

How an Operating System Manages an Application

An application depends on an OS to provide access to hardware resources, manage its data in memory and in secondary storage, and perform many other background tasks. Earlier OSs gave applications more latitude and power than later OSs do. For example, under DOS, a program could write data directly to RAM, but now, under Windows NT and Windows 2000, an application is not allowed direct access to RAM. Using today's OSs, the interaction among the hardware, the OS, and applications is more complicated than it was under DOS.

This book focuses on Microsoft operating systems: DOS, Windows 9x, Windows NT, and Windows 2000, by far the most popular OSs for microcomputers. It's true that DOS has become less popular as an OS in recent years. But because Windows has its beginnings rooted in DOS, it is important to understand DOS and the early assumptions and limitations of this OS that still affect Windows today. The two concepts in DOS discussed here are the rules DOS uses to name files (referred to as naming conventions) and how DOS manages memory addressing. Then, in leading up to discussing how applications software interacts with an OS today, we address how DOS interacts with a DOS application. The knowledge gained here will serve you well as you study how Windows 9x accomplishes these same tasks.

A+ OS 1.2

Filenames Under DOS Under DOS, a file's name has two parts. The first part, called the **filename**, contains up to eight characters. The second part, called the **file extension**, contains up to three characters. When you write the file extension in DOS commands, you separate the extension from the filename with a period. Acceptable file extensions for program files are .com, .bat, and .exe. For example, the WordPerfect program file is named WP.exe. Its filename is WP and its file extension is .exe.

With the introduction of Windows 95, long filenames traditionally used only by the Macintosh OS became available to IBM-compatible PCs. Under Windows 95 and later Windows generations, filenames can be as long as 255 characters and may contain spaces. You must be careful when using long filenames with Windows 9x because Windows 9x still contains a portion of DOS, which can only understand an 8-character filename, 3-character extension format. When the DOS part of the system is operating, it will truncate long filenames and assign new 8-character ones. More information about naming files appears in Chapter 5.

Memory Addressing Under DOS In DOS, memory is divided into different areas, as shown in Table 1-9. This division of memory began with DOS and later was used by Windows 3.x and Windows 9x. The first versions of DOS could only access the first megabyte of memory (0 to 1024K) addresses. Later, DOS extenders were included with the OS so that memory above 1024K could be accessed.

Table 1-9 Division of memory under DOS

Range of Memory Addresses	Type of Memory
0 to 640K	Conventional or base memory
640K to 1024K	Upper memory
Above 1024K	Extended memory

Operating System Modes Recall that an application, the OS, and BIOS can share data by communicating memory addresses, which point to the location of the data in RAM (see Figure 1-29). Also recall that memory addresses are communicated on the system board over a bus where a group of wires, or lines, in the bus are dedicated to this purpose. When DOS was first written, the system board bus only had 20 lines for this purpose. Since information is communicated over the bus in binary (0s and 1s), the largest memory address possible for these system boards was 1111 1111 1111 1111 1111 in binary, (2^20) which is 1,048,575 in decimal (see Appendix D for an explanation of this conversion). In effect, there were 1024K of possible memory addresses, and that was all that DOS was designed to use. Also, DOS assumed that only one application or program would be running at a time, so it gave that program direct access to these memory addresses and the data in RAM that they pointed to. Also, the CPU at that time (Model 8088 by Intel Corporation) could only process 16 bits of data at one time, so DOS was designed to pass segments of data to the CPU only 16 bits at a time. These standards of operation are collectively referred to as **real mode**, or MS-DOS mode, as seen in Figure 1-36.

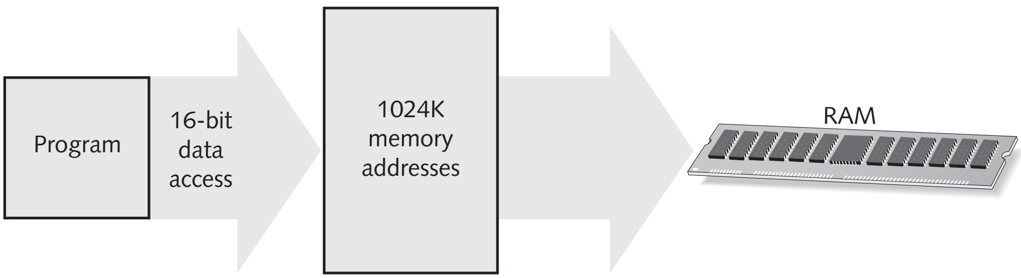

Figure 1-36 Real mode is single-tasking program access to 1024K of memory addresses pointing directly to RAM using a 16-bit data path

TIP

Real mode is limited to 16-bit data processing because originally only 16 lines on the system board bus were devoted to transmitting data to and from the CPU.

Later CPUs and system boards had buses that had as many as 32 wires devoted to memory addresses, allowing up to 4096 MB of addresses and data paths that were 32 bits wide. Both the CPU and the OS could support cooperative multitasking. These new operating methods are known as **protected mode** because the OS ensures that the memory assigned to one program is protected from interference by other programs. Figure 1-37 shows that, in protected mode, more than one program can run, and the programs have access to memory addresses up to 4096 MB (depending on the system board, CPU, and OS being used).

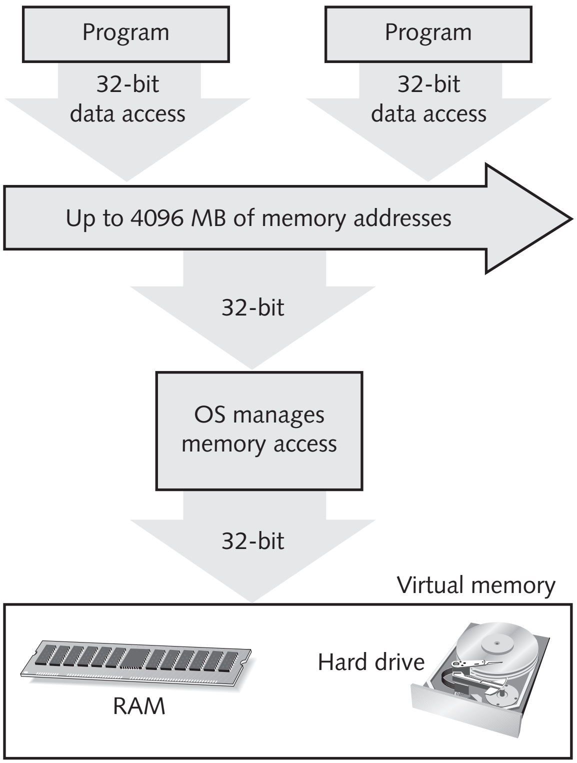

Figure 1-37 Protected mode is multitasking program access to more than 1024K of memory addresses using 32-bit data segments where the OS manages direct memory access

In protected mode, the OS does not allow a program direct access to RAM, but works as the mediator between memory and programs. This allows the OS some latitude in how it uses RAM. If the OS is low on RAM, it can store some data on the hard drive. This method of using the hard drive as though it were RAM is called **virtual memory**, and data stored in virtual memory is really stored in a file on the hard drive called a **swap file**. The OS manages the entire process, and the applications know nothing about this substitution of hardware resources. The programs running in protected mode just have "memory" designated by memory addresses and have no idea where it's located.

Even after protected mode was available, it was important for hardware and software to be **backward compatible** (able to support older technology), so real mode was still supported by the CPUs and OSs. In fact, most OSs start in real mode and switch to protected mode before allowing user interaction. DOS operates in real mode. All DOS programs use a 16-bit interface with the CPU, but can use memory addresses above 1024K by using a DOS extender, which will be discussed in detail in later chapters. Windows 9x starts out in real mode and then switches to protected mode. Some of Windows 9x uses 16-bit data access (called 16-bit programs) and some uses 32-bit data access (called 32-bit programs). Because the 32-bit programs access twice as much data at one time as 16-bit programs do, the 32-bit programs are faster. This fact largely explains why Windows 9x is faster than DOS or DOS with Windows 3.x. Windows NT is a true 32-bit OS; all its OS programs are written using 32-bit coding methods.

TIP The main reason Windows 9x is faster than DOS or DOS with Windows is because it uses 32-bit data access, rather than the slower 16-bit access used by the older Windows software. However, Windows 9x still supports 16-bit software.

An OS will allow a 16-bit program to run in a 32-bit environment by providing it with an environment that appears to the program to be a 16-bit environment. This technique is called **virtual real mode** and will be discussed in Chapter 2.

Applications Software

Most applications software fits into eight categories: word processing, spreadsheet, database management, graphics, communications, games, mathematical modeling, and software development tools. Each software category contains many different products. For example, some popular database management packages include Access, Paradox, and Filemaker, and two popular word-processing packages include Word and WordPerfect. Some applications software manufacturers produce **suites** of software, which combine a word-processing program and spreadsheet program, and usually include a database management program, a presentation package, an e-mail package, and a World Wide Web browser package. Suites provide many advantages, including letting the programs use the same basic instruction sets; the programs are designed to make it easy to move data from one suite program to another; and files within a suite's programs can be linked, so that updates to data or text are automatically recorded in all linked files.

Applications software is designed to work on top of a particular OS. "On top of" here means that the application depends on the OS, such as MS-DOS or Windows 98, in order to run. For example, consider a situation in which Windows 98 loads an application and executes it. The application cannot run or even load itself without Windows 98, much as a document cannot be edited without a word-processing program. Windows 98 stays available to the application for the entire time the application is running. The application passes certain functions to Windows 98, such as reading from a CD-ROM or printing.

An application written to work with one OS, such as Windows 98, does not necessarily work with another, such as a Macintosh system. There are, however, some exceptions. For instance,

OS/2 is written so that any application designed to work with DOS also works with OS/2, an excellent early selling point for OS/2. However, to take full advantage of an OS's power and an application's power, buy applications software written specifically for your OS.

A+ OS 2.4

Applications software comes written on floppy disks or on CD-ROMs and usually must be installed on a hard drive in order to run. Installing a software package usually is very easy. For instance, under DOS, typically you insert the first disk of a set of floppies or a CD-ROM, and then type a command, such as A:INSTALL or D:SETUP at the DOS prompt. If you are working in Windows 9x, you click the Start button, click Run, and then follow the directions on the screen. We discuss software installation in Chapter 12.

How Applications Software Is Loaded and Initialized

Starting applications software seems like a simple task. For example, in Windows 98, you click the shortcut icon on the desktop, and the application window appears, ready for you to use it. But much happens behind the scenes between your click and the appearance of the application awaiting your next command. Recall that software or programs are stored in program files until needed. Before they can be executed, they must first be copied or loaded into RAM and assigned memory addresses. The CPU then works its way through RAM, following each instruction in turn. If the program uses data (and most do), the data is permanently stored in a data file in secondary storage, and must be copied or loaded into RAM and assigned memory addresses before the CPU can process it (see Figure 1-38). Understanding the process of loading and initializing software is important to those people responsible for supporting PCs. Listed below are the major steps that must take place. This section describes the process in detail for several OSs.

1. The OS receives the command to execute the application.
2. The OS locates the program file for the application.
3. The OS loads the program file into memory.
4. The OS gives control to the program.
5. The program requests memory addresses from the OS for its data.
6. The program initializes itself and possibly requests that data from secondary storage be loaded into memory.
7. The program turns to the user for its first instruction.

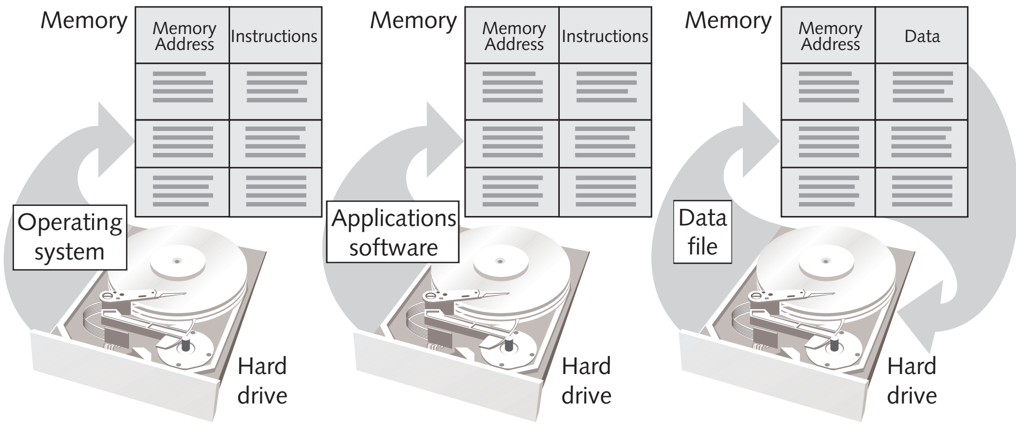

Figure 1-38 Both the operating system and applications software must be loaded into memory before execution, and existing data must also be copied from secondary storage into memory to be processed. After processing, data must be saved back to secondary storage.

We begin with DOS as our OS. Before we discuss the specific command to load software in DOS, let's first look at the command prompt the user sees and what information it provides. If DOS is your OS, the DOS prompt displays basic information and gives you the opportunity to enter some command for the OS to perform. Most of the time, DOS is loaded from the hard drive, which is designated with the letter C. It is common for computers to provide a DOS prompt that looks like this when the machine is first turned on:

```
C:\>
```

This prompt is called the C prompt. The DOS prompt (C:\>) displayed immediately after booting means that the OS was copied from drive C (where the OS is stored) when the machine was first turned on. As part of the startup process, drive C then becomes the **default drive** and **default directory**, sometimes called the current drive and directory, which DOS automatically uses to save and retrieve files. Sometimes the OS is copied from a floppy disk rather than the hard drive. In this case, the default drive and directory will be A:\ or B:\, and the command prompt will be A:\> or B:\>. A machine usually has at least two drives. The colon following the letter identifies the letter as the name of a drive, and the backslash identifies the directory on the drive as the main directory. The > symbol is the prompt symbol that DOS uses to say, "Enter your command here."

When a hard drive is first formatted for use by an OS, the format procedure creates a single directory on the drive. A directory is a table on a disk that contains a list of files that are stored on the disk. You can think of a directory as a list of files logically grouped together. When a hard drive is first formatted, the **root directory** (that is, the first or main directory on a drive) is the only directory that is created. This directory is written in DOS command lines as a single backslash (\) with no other directory name following. In the preceding DOS prompt, the backslash indicates that it is the root directory. After the drive is initially formatted, you can create other directories for file lists. These directories are given names and are listed in the root

directory. These directories can, in turn, have other directories listed in them. These other directories are sometimes called **subdirectories** or **child directories** (see Figure 1-39). Any directory can have files and/or other directories listed in it. By creating different directories on a hard drive, you can organize your program files and data files by placing programs in one directory, and files created by those programs in a second directory. This organization is comparable to keeping paper records in separate folders.

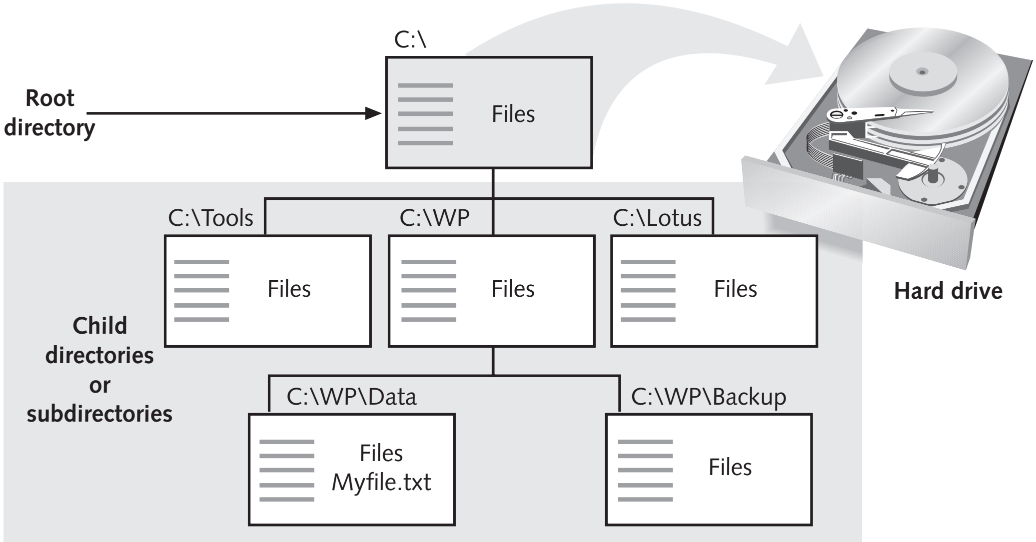

Figure 1-39 A hard drive is organized into groups of files stored in directories. The first directory is called the root directory. All directories can have sub- or child directories. Under Windows, a directory is called a folder.

A drive with a list of directories pointing to a file is called a **path** when used in a DOS command. For instance, if the text file Myfile.txt is stored in the Data directory under the WP directory on the hard drive, the path to Myfile.txt is written as C:\WP\Data\Myfile.txt. The path is used to find the file. For instance, to see the contents of the text file Myfile.txt from any DOS prompt, we use the TYPE command, like this:

```
TYPE C:\WP\DATA\MYFILE.TXT
```

The TYPE command looks on drive C for a directory named WP. In that directory it looks for a subdirectory named DATA. DOS expects to find the file inside this subdirectory. It is appropriate to say that Myfile.txt is located in the path C:\WP\Data. If the file is located anywhere else on the hard drive, DOS will not find it, because we have given it only this one path to the file.

Launching the Program File Now that you understand the DOS prompt and a path to a file, we turn our attention to giving DOS a command to execute a program. At the DOS prompt, when you type a single group of letters with no spaces, DOS assumes that you want to execute a program with the filename that you just typed, stored in a program file in the

current directory. DOS first attempts to find the program file by that name, then copies the file into RAM, and then it executes the program. Let's use running WordPerfect in DOS as an example. In order to run WordPerfect in DOS, you type the letters WP when the OS displays the DOS prompt:

```
C:\>WP
```

A program file executed at the DOS prompt can have one of three file extensions: .com, .exe, or .bat. Therefore, in this example, DOS looks for a file with one of these names: WP.com, WP.exe, or WP.bat.

A+ OS 1.1

The first place DOS looks is in the default drive and directory—in this case, the root directory of the hard drive. If the A:\> prompt was displayed when you typed WP, then DOS would look in the root directory of drive A for the program file. DOS looks for the files in the following order: first WP.com, then WP.exe, and finally WP.bat. DOS executes the first one it finds. If DOS doesn't find any of these files, it stops looking and displays the error message:

```
Bad command or file not found
```

unless you have previously given DOS a list of paths in which to look for executable program files beyond the default directory. You give this list of drives and directories to DOS using the PATH command. You can cause the PATH command to be executed automatically during the booting process by storing the command in the Autoexec.bat file (to be discussed next). However, you can execute the PATH command at any time after booting. The last PATH command you execute overrides any previous one. To see the list of paths that are presently active, type PATH at the DOS prompt, and then press Enter. To enter a new list of paths, type the PATH command followed by each path name, separating one path from the next by a semicolon, as shown in Figure 1-40.

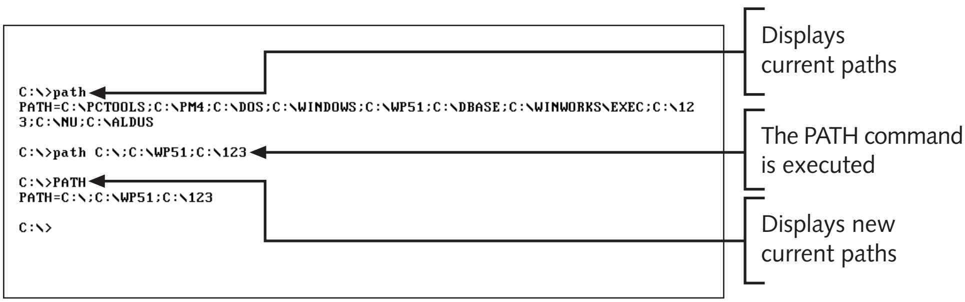

Figure 1-40 The PATH command

In Figure 1-40, the first PATH command displays the list of active paths. The second PATH command changes this list, giving DOS three directories in which to look to find executable files:

- Hard drive and root directory (C:\)
- Hard drive and directory named \WP51 (C:\WP51)
- Hard drive and directory named \123 (C:\123)

A+ OS 1.1

The last PATH command in Figure 1-40 again displays the list of active paths.

When you tell DOS to execute a program, you can also include the path to that program file as part of the command line. For example, if the WP.exe file is stored in the directory \WP51, you can execute the program by typing the following:

```
C:\> \WP51\WP
```

Here you are telling DOS that the name of the program file is WP and that its location is in the directory \WP51. DOS assumes that the directory \WP51 is on the hard drive rather than on drive A or B, because C is the current default drive. With this method, the directory that contains WordPerfect need not be the default directory, nor do you need the \WP51 path in the PATH command line in the Autoexec.bat file. DOS used Rule 3 in the rules listed earlier for finding a program file.

Another way that you can tell DOS where the program is located is to make the directory containing the program file the current default directory, by using the CHANGE DIRECTORY, or CD, command. For example, you can use the following two commands to execute WordPerfect:

```
C:\> CD \WP51
C:\WP51> WP
```

The CD command directs DOS to make the directory \WP51 the default directory. This default directory appears as part of the DOS prompt. The next command executes the WordPerfect program stored in this directory. Because \WP51 is now the default directory, unless you direct WordPerfect otherwise, documents are saved to and retrieved from the \WP51 directory.

In summary, DOS searches for executable program files using the following rules:

1. If no path is given before the filename, DOS looks in the current directory.
2. If no path is given, and the file is not in the current directory, DOS looks in the paths given to it by the last PATH command executed.
3. If there is a path given in front of the filename in the command line, DOS looks in that path.
4. If there is a path given, but the file is not found in that path, DOS looks in the paths given to it by the last PATH command executed.

From this discussion, you can see that when you tell DOS to execute a program, DOS is unable to find that program unless you do one of the following:

- Include the path to the program file in the PATH command in Autoexec.bat
- Make the directory that contains the program file the default directory, using the CD command
- Include the path to the program file in the command line to execute the program

Copying the Program into Memory Recall that once DOS finds the program file, it copies the file into memory (RAM) in a location that DOS chooses (see Figure 1-41). After it copies the program into memory, DOS goes to the first address in memory occupied by the program, to receive its first instruction. If the program requests some memory for its data (and most will), DOS decides which memory addresses to give the program (usually the memory after the program).

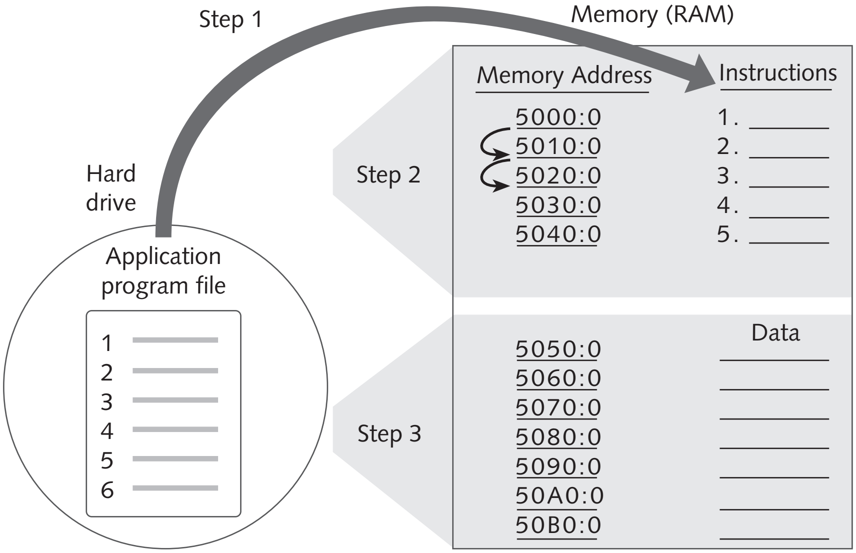

Figure 1-41 Applications software is stored in files but executed from memory

If the program wants to write or read data to or from memory, DOS manages these tasks. If the program needs to print, display something on the screen, or read or write to or from the hard drive or a floppy disk, DOS does the work and returns to the applications software when finished. In other words, DOS is the "software behind the software," doing the background tasks for the application.

Loading Applications Software Using Windows 9x

Windows 9x has a different interface and methods to execute software than does DOS. Windows 9x offers four ways to execute software:

- Place a shortcut icon directly on the desktop for the applications you use often and want to get to quickly. These shortcuts contain the command line used to execute the application. To view this command line, right-click on an icon. A drop-down menu then displays. From the menu, select Properties. The icon's Properties box displays. See Figure 1-42. From this box, you can view the complete command line that the icon represents.
- Click the Start button, select Programs, and select the program from the list of installed software.

- Use the Run command: Click the Start button on the Windows 9x taskbar and then click Run to display the Run dialog box. See Figure 1-43. In this box, enter a command line or click Browse to search for a program file to execute.
- Execute a program or launch an applications file by double-clicking the filename in Windows 9x Explorer or My Computer.

A+ OS 1.1

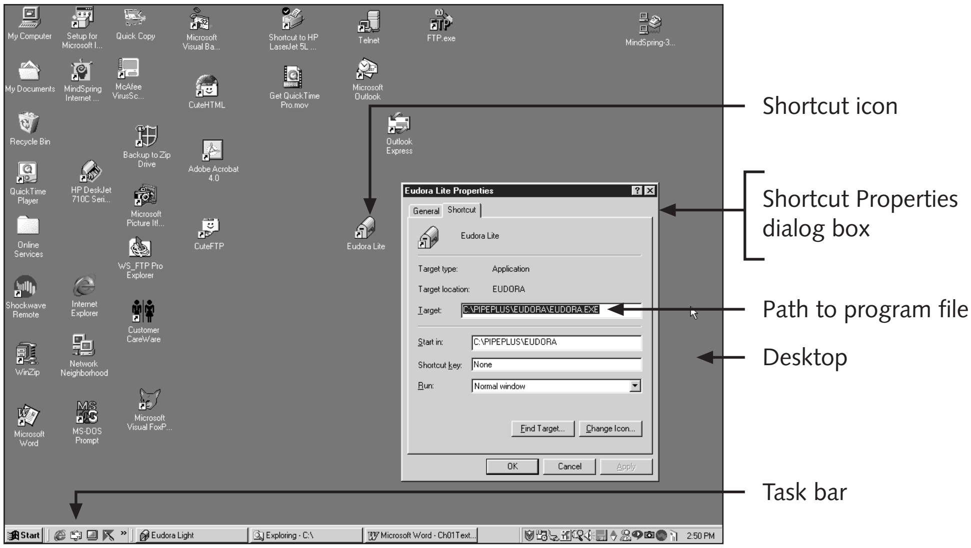

Figure 1-42 Windows 9x has icons on the desktop that point to program files on the hard drive

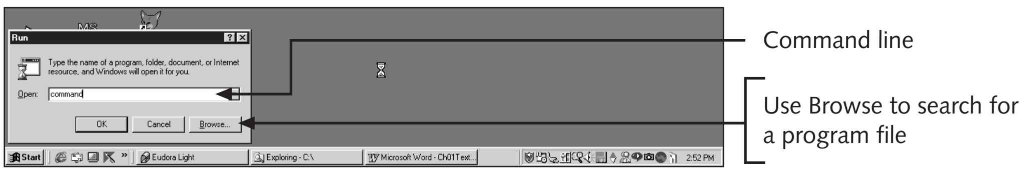

Figure 1-43 The Windows 9x Run dialog box allows you to enter DOS-like commands

In summary:

- Applications software is executed by either the operating environment (Windows 3.x) or the OS software (DOS or Windows 9x).
- When an application is executing, you are interacting with the application.
- Applications software interacts with the OS software that is executing it.
- OS software interacts with hardware.
- OS software might interact with the hardware through BIOS or the device driver for this hardware.

CHAPTER SUMMARY

- The four basic functions of the microcomputer are input, output, processing, and storage of data.
- The four most popular input/output devices are the printer, monitor, mouse, and keyboard.
- The most important component inside the computer case is the system board, or motherboard, which contains the most important microchip inside the case, the central processing unit (CPU), a microprocessor, as well as access to other circuit boards and peripheral devices. All communications between the CPU and other devices must pass through the system board.
- Data and instructions are stored in a computer in binary, which uses only two states for data, on and off or 1 and 0.
- A ROM BIOS microchip is a hybrid of hardware and software containing programming embedded into the chip. These chips are called firmware.
- Each hardware device needs a method to communicate with the CPU, software to control it, and electricity to power it.
- Devices outside the computer case are connected to the system board by way of ports on the case.
- A circuit board inserted in an expansion slot on the system board can be used to provide an interface between the system board and a peripheral device, or can itself be a peripheral (an example is an internal modem).
- The chip set on a system board relieves the CPU of some of the system's processing demands and controls many components on the board.
- Primary storage, or RAM, is temporary storage used by the CPU to hold data and instructions while it is processing both.
- RAM is stored on single chips, SIMMs, DIMMs, and RIMMs.
- Cache memory is used on the system board or inside the CPU housing as fast RAM to improve processing speed.
- The buses on the system board are used to communicate data, instructions, and electrical power to components on the board.
- The system clock is used to synchronize activity on the system board, sending continuous pulses over the bus to different components.
- Setup or configuration information in a PC is stored in the CMOS chip on the system board and by means of jumpers and switches.
- Electricity is supplied to components both inside and outside the computer case by the power supply inside the case. Some components external to the case get their power from their own electrical cable.

- Secondary storage is slower than primary storage, but is permanent storage. The most common examples of secondary storage devices are floppy disk, hard drive, and CD-ROM drive.
- Three types of software are BIOS, operating systems (OSs), and applications software.
- BIOS is used at startup and after startup by the OS to provide software control of hardware devices.
- Applications software relates to the OS, which relates to BIOS and device drivers to control hardware.
- Operating systems utilize BIOS, manage secondary storage and primary storage, help diagnose problems with hardware and software, interface between hardware and software, and perform various housekeeping tasks.
- When a PC is first turned on, the startup BIOS is in control. It later loads the OS and then turns control over to it.
- Users interact with an OS by a command-driven, menu-driven, or icon-driven interface.
- The most well-known OSs for microcomputers are DOS, Windows, UNIX, Mac OS, and OS/2.
- True multitasking is not possible using CPUs built before the Pentium.
- DOS is being replaced by Windows as the most popular OS, but decisions made when DOS was designed still affect Windows 9x today.
- For DOS, the three types of logical primary memory are conventional (or base) memory, upper memory, and extended memory.
- Software manages memory by means of memory addresses that point to locations in RAM. The number of memory addresses is partly limited by the number of wires on the bus devoted to these addresses.
- The size of the data segment that software can access at one time is determined by the number of wires on the bus assigned for the data path.
- Real mode was used by DOS and is limited to single-tasking, a 16-bit data path, and 1024K of memory addresses.
- Protected mode allows more than one program to run at a time, can use a 32-bit data path, and has more than 1024K of memory addresses. In protected mode, the OS manages access to RAM and does not allow a program direct access to it.
- Virtual memory is "fake" memory whereby data is stored in a swap file on the hard drive. The OS makes applications think that they are using real memory.
- When an OS receives the command to execute a software program, it follows explicit rules as to where it looks to find the program file for the software.
- A program must first be loaded into memory before the OS can execute it.

Key Terms

Backward compatible — Refers to new hardware and software that is able to support older, existing technologies. This is a common choice of hardware and software manufacturers.

Binary number system — The number system used by computers; it has only two numbers, 0 and 1, called binary digits, or bits.

BIOS (basic input/output system) — Firmware that controls much of a computer's input/output functions, such as communication with the floppy drive, RAM chips, and the monitor. Also called RAM BIOS.

Booting — The process that a computer goes through when it is first turned on to get the computer ready to receive commands.

Bus — The paths, or lines, on the system board on which data, instructions, and electrical power travel.

Cache memory — A kind of fast RAM that is used to speed up memory access because it does not need to be continuously refreshed.

Cards — Adapter boards or interface cards placed into expansion slots to expand the functions of a computer, allowing it to communicate with external devices such as monitors or speakers.

Child directory — *See* Subdirectory.

Chip set — A group of chips on the system board that relieves the CPU of some of the system's processing tasks, increasing the overall speed and performance of the system.

Circuit boards — Computer components, such as the main system board or an adapter board, that have electronic circuits and chips.

Client — A computer that is connected to another computer and uses programs and/or data stored on the other computer.

CMOS (complementary metal-oxide semiconductor) — One of two types of technologies used to manufacture microchips (the other type is TTL, or transistor-transistor logic chips). CMOS chips require less electricity, hold data longer after the electricity is turned off, are slower, and produce less heat than do TTL chips. The configuration or setup chip is a CMOS chip.

COAST (cache on a stick) — Memory modules that hold memory used as a memory cache. *See* Cache memory.

Cooperative multitasking — A type of pseudomultitasking whereby the CPU switches back and forth between programs loaded at the same time. One program sits in the background waiting for the other to relinquish control. Also called task switching.

Coprocessor — A chip or portion of the CPU that helps the microprocessor perform calculations and speeds up computations and data manipulations dramatically.

CPU (central processing unit) — Also called a microprocessor or processor. The heart and brain of the computer, which receives data input, processes information, and executes instructions.

Default directory — The directory that DOS automatically uses to save and retrieve files.

Default drive — The drive that DOS automatically uses to save and retrieve files.

Default printer — The printer that Windows software will use unless the user specifies another printer.

Desktop — The initial screen that is displayed when an OS that has a GUI interface is loaded.

Device driver — A small program stored on the hard drive that tells the computer how to communicate with an input/output device such as a printer or modem.

DIMM (dual inline memory module) — A miniature circuit board used in newer computers to hold memory. DIMMs can hold 16, 32, 64, or 128 MB of RAM on a single module.

DIP (dual in-line package) switch — A switch on a circuit board or other device that can be set on or off to hold configuration or setup information.

Environment — As related to OSs, the overall support that an OS provides to applications software.

Expansion card — A circuit board inserted into a slot on the system board to enhance the capability of the computer.

Expansion slot — A narrow slot on the system board where an expansion card can be inserted. Expansion slots connect to a bus on the system board.

File — A collection of related records or lines that can be written to disk and assigned a name (for example, a simple letter or a payroll file containing data about employees).

File extension — A three-character portion of the name of a file that is used to identify the file type. The file extension follows the filename under DOS naming conventions.

Filename — The first part of the name assigned to a file. In DOS, the filename can be no more than eight characters long and is followed by the file extension.

Firmware — Software that is permanently stored in a chip.

Flash ROM — ROM that can be reprogrammed or changed without replacing chips.

GUI (graphical user interface) — A user interface, such as the Windows interface, that uses graphics or icons on the screen for running programs and entering information.

Hard copy — Output from a printer to paper.

Hard drive — The main secondary storage device of a PC, a sealed case that contains magnetic coated platters that rotate at high speed.

Hard drive controller — A set of microchips with programs that control a hard drive. Most hard drive controllers today are located inside the hard drive housing.

Hardware — The physical components that constitute the computer system, such as the monitor, the keyboard, the system board, and the printer.

Jumper — Two wires that stick up side by side on the system board that are used to hold configuration information. The jumper is considered closed if a cover is over the wires, and open if the cover is missing.

Keyboard — A common input device through which data and instructions may be typed into computer memory.

Main board — *See* System board.

Memory address — A number assigned to each byte in RAM. The CPU can use memory addresses to track where information is stored in RAM. Memory addresses are usually displayed as hexadecimal numbers in segment/offset form.

Monitor — The most commonly used output device for displaying text and graphics on a computer.

Motherboard — *See* System board.

Mouse — A pointing and input device that allows the user to move a cursor around a screen and select programs with the click of a button.

Multitasking — When a CPU or an OS supporting multiple CPUs can do more than one thing at a time. The Pentium is a multitasking CPU.

Nonvolatile — Refers to a kind of RAM that is stable and can hold data as long as electricity is powering the memory.

On-Board BIOS — *See* System BIOS.

Path — The drive and list of directories pointing to a file.

Peripheral devices — Devices that communicate with the CPU, but are not located directly on the system board, such as the monitor, floppy drive, printer, and mouse.

Pixel — The smallest dot that can be addressed by software on a monitor screen. An image is composed of many pixels.

Plug and Play — A feature of system BIOS, Windows 9x, and Windows 2000 that automatically installs new hardware devices and assigns resources to them.

Port — A physical connector, usually at the back of a computer, that allows a cable from a peripheral device, such as a printer, mouse, or modem, to be attached.

Power supply — A box inside the computer case that supplies power to the system board and other installed devices. Power supplies provide 3.3, 5, and 12 volts DC.

Preemptive multitasking — A type of pseudomultitasking whereby the CPU allows an application a specified period of time and then preempts the processing to give time to another application.

Primary storage — Temporary storage on the system board used by the CPU to process data and instructions.

Printer — A peripheral output device that produces printed output to paper. Different types include dot matrix, ink-jet, and laser printers.

Program — A set of step-by-step instructions to a computer. Some are burned directly into chips, while others are stored as program files. Programs are written in languages such as BASIC and C++.

Program file — A file that contains instructions designed to be executed by the CPU.

Program jump — An instruction that causes control to be sent to a memory address other than the next sequential address.

Protected mode — An operating mode that supports multitasking whereby the OS manages memory, programs have more than 1024K of memory addresses, and programs can use a 32-bit data path.

Protocol — A set of rules and standards that two entities use for communication.

RAM (random access memory) — Temporary memory stored on chips, such as SIMMs, inside the computer. Information in RAM disappears when the computer's power is turned off.

Real mode — A single-tasking operating mode whereby a program only has 1024K of memory addresses, has direct access to RAM, and uses a 16-bit data path.

RIMM — A type of memory module used on newer system boards

ROM (read-only memory) — Chips that contain programming code and cannot be erased.

ROM BIOS — *See* BIOS.

Root directory — The main directory created when a hard drive or disk is first formatted.

Secondary storage — Storage that is remote to the CPU and permanently holds data, even when the PC is turned off.

Server — A microcomputer or minicomputer that stores programs and data to be used remotely by other computers.

SIMM (single inline memory module) — A miniature circuit board used in a computer to hold RAM. SIMMs hold 8, 16, 32, or 64 MB on a single module.

Software — Computer programs, or instructions to perform a specific task. Software may be BIOS, OSs, or applications software such as a word-processing or spreadsheet program.

Startup BIOS — Part of system BIOS that is responsible for controlling the PC when it is first turned on. Startup BIOS gives control over to the OS once it is loaded.

Subdirectory — In DOS, a directory that is contained within another directory. Also called a child directory.

Suite — As applies to applications software, a collection of applications software sold as a bundle, whose components are designed to be compatible with one another. An example is Microsoft Office.

Swap file — A file on the hard drive that is used by the OS for virtual memory.

System BIOS — BIOS located on the system board.

System board — The main board in the computer, also called the motherboard. The CPU, ROM chips, SIMMs, DIMMs, and interface cards are plugged into the system board.

System clock — A line on a bus that is dedicated to timing the activities of components connected to it. The system clock provides a continuous pulse that other devices use to time themselves.

Task switching — *See* Cooperative multitasking.

Trace — A wire on a circuit board that connects two components or devices.

Video card — An interface card installed in the computer to control visual output on a monitor.

Virtual memory — A method whereby the OS uses the hard drive as though it were RAM.

Virtual real mode — An operating mode in which an OS provides an environment to a 16-bit program that acts like real mode.

Volatile — Refers to a kind of RAM that is temporary, cannot hold data very long, and must be frequently refreshed.

REVIEW QUESTIONS

1. Why is all data stored in a computer in binary form?
2. How does firmware differ from a software program stored on the hard drive?
3. What are the four primary functions of hardware?
4. What three things do electronic hardware devices need in order to function?
5. Name two input devices.
6. Obtain the manual for a system board and look for a diagram of the system board components similar to Figure 1-8. Identify as many components from the figure on your diagram as you can.
7. What is the purpose of an expansion slot on a system board?
8. Which components on the system board are used primarily for processing?
9. Which can hold the most memory, a SIMM or a DIMM?
10. What three things travel on a system board bus?
11. Give three examples of secondary storage devices.
12. Which OS (include version) does your home or lab computer use?
13. What does BIOS stand for, and what does it do?
14. List three well-known OSs.
15. Give three examples of OSs that use a GUI.
16. What might cause the error message "Bad command or file not found"?
17. What is the default directory of your home or lab computer immediately after bootup?
18. In the chapter, the concepts of primary storage and secondary storage are introduced. In Figure 1-38, a program was moved from secondary storage to primary storage before it could be executed. Based on this figure and concept, why do you think a hard drive is called secondary storage and memory is called primary storage?
19. Complete the following table. Refer to Appendix D as necessary.

Type of Memory	Beginning Memory Address in Hex	Ending Memory Address in Hex	Beginning Memory Address as a Decimal Number	Ending Memory Address as a Decimal Number	Ending Memory Address in Decimal Kilobytes
Conventional or base	0	9FFFF			
Upper	A0000	FFFFF			
Extended	100000	N/A		N/A	N/A

20. The number of memory addresses is limited by the ________________ on the system board.
21. The amount of memory available to software is limited by the ________________ on the system board.
22. In what three ways can configuration information be stored on a system board?

23. Name one way BIOS and device drivers are the same. Name one way they are different.
24. When one software layer passes data to another, it usually refers to the ________________ of the data.
25. How does cooperative multitasking differ from preemptive multitasking?
26. Which OS was the first to introduce Plug and Play?
27. Using DOS, why is this not a valid file name?: EMPLOYEES.TXT
28. Memory above 1024K is called ________________.
29. Real mode operates using a ________________-bit data path, and protected mode uses a ________________-bit data path.
30. Real mode allows programs direct access to ________________, but protected mode does not.

PROJECTS

Observing the Boot Process and Hardware Components

1. Carefully watch your computer screen during the boot process (press Pause if necessary), and record which CPU is used by your home or lab computer.
2. Who is the BIOS vendor and what version of the BIOS are you using?
3. As the computer boots, memory is counted. Observe the memory count and record the amount of memory detected. What number system is used to count this memory?
4. Open the printer icon in the Windows Control Panel and find out which is the default Windows printer for your home or lab computer.
5. Look at the back (or the front if the ports are located there) of your home or lab computer and make a drawing. Label on the drawing the purpose of each port and connection you see. If you are not sure what the purpose of the port is, label the port "unknown port." In later chapters, the purposes of these unknown ports will become clear.

Using the Internet for Research

Linux is one of the fastest growing OSs today. What is the official mascot for Linux? Who chose it, and why? *Hint*: See *www.linux.org*.

Using Microsoft Diagnostics with Windows

DOS and Windows offer the Microsoft Diagnostics command. This utility examines your system, displaying useful information about ports, devices, memory, and the like. (For Windows 9x, search for the MSD.EXE utility on your Windows 9x installation CD and copy it to your hard drive.) Boot your PC to a DOS prompt. From the DOS prompt, execute this command:

```
C:\> MSD
```

You should see a screen similar to that in Figure 1-44. Browse carefully through all the menu options of this interesting utility and answer the following questions about your system:

a. List the following or print the appropriate MSD screen:

   Manufacturer, version number, and date of your System BIOS, video BIOS, and mouse device driver.

b. What kind of video card is installed?

Use the information in Appendix D, "The Hexadecimal Number System and Memory Addressing," to answer these questions:

- How much memory is currently installed on this PC?
- Look under TSR programs (terminate-and-stay-resident programs, programs currently stored in memory but not running, will be covered in a later chapter) for the MSD.exe program that you are executing. What is the hex address of the beginning of this program? Convert the hex address to a decimal address.
- What version of DOS are you running?
- What CPU are you using?

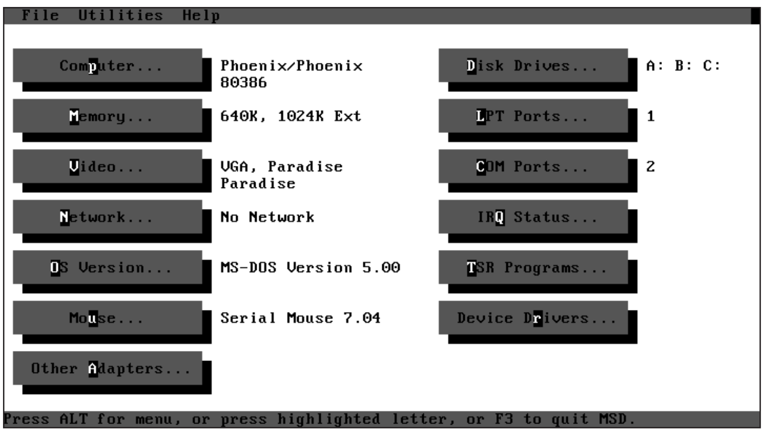

Figure 1-44 MSD opening screen

When you are finished, exit the MSD utility and return to the C prompt.

Using Device Manager

Windows 9x provides a much more powerful tool than MSD, called Device Manager. From it you can view and print your hardware configuration. To access Device Manager, follow these steps.

1. Click the **Start** button on the taskbar, click **Settings**, and then click **Control Panel**.
2. When the Control Panel window appears, double-click the **System** icon.
3. In the System Properties dialog box, click the **Device Manager** tab.

The opening menu of Device Manager appears as shown in Figure 1-45. You can select an item from the list and then click Properties to view information about that item, or you can view the Properties by double-clicking the item. When you click the + sign to the left of an item, a list of the installed devices for that item appears beneath the item.

Figure 1-45 The Device Manager in Windows 98

Answer these questions about your computer:

1. Does your computer have a network card installed? If so, what is the name of the card?
2. What are three settings that can be changed under Device Manager?
3. What are all the hardware devices that Device Manager recognizes as present?

Use Shareware to Examine a Computer

Note: This exercise requires access to the Internet.

Good PC support people are always good investigators. The Internet offers a wealth of resources to those who take the time to search, download, and investigate the possible uses

of software available there. This exercise is designed to help you learn to be such an investigator. Follow these directions to download a shareware utility to diagnose Windows 9x or Windows 3.x problems and print a report from the downloaded software about the hardware and software on your computer.

1. Access the Internet and go to this address:

   http://www.zdnet.com

2. Search for SANDRA. SANDRA stands for System Analyzer Diagnostic and Reporting Assistant and offers information about the hardware and software on your computer.
3. Follow the steps on the screen to download the file Sandra.zip to your PC. You can then disconnect from the Internet.
4. Uncompress Sandra.zip by double-clicking the filename and then extracting Setup.exe with its components.
5. Run the setup program, Setup.exe, which creates a new program in your Program Group.
6. Run the program SiSoft Sandra, and the screen shown in Figure 1-46 appears. You can execute each of the icons in turn by double-clicking them, or you can create a composite report of the results of each selection. Answer these questions:
   a. What is the model and speed of your CPU?
   b. Which version of Windows are you using?
   c. Which icons are not available to you because your copy of SANDRA is not registered?

Figure 1-46 Shareware utility for Windows 9x

You will use SANDRA again in later chapters, so don't erase her! By the way, try to find SANDRA through *www.shareware.com* and *www.sisoft.com*. Is the program available through these avenues as well?

Using Nuts & Bolts to Examine Your System

Follow the directions in the Preface to install Nuts & Bolts to your hard drive from the CD-ROM accompanying this book. Use Discover Pro to answer the following questions. (To access Discover Pro, click **Start**, **Programs**, **Nuts & Bolts**, **Discover Pro**.)

1. What CPU are you using?
2. What CPU speed does the manufacturer claim?
3. At what actual speed is the CPU running?
4. What is the BIOS manufacturer and production date?
5. How much RAM is in the system?
6. Which version of Windows are you using? Which version of DOS?
7. What software is currently running (Discover Pro calls these "tasks")? Are they 16-bit or 32-bit programs?
8. What is the size of your hard drive? How much of the drive is in use? How much is free?
9. What printers are installed on the computer? (An installed printer means that Windows has the printer device driver installed and available for use.)
10. Under Discover Pro Diagnostics, run the test of memory. Were any memory errors discovered?